

TRAFFIC SHAPING NETWORKS

By

Maninder Pal Singh

(B.Tech, Punjab Technical University, Jalandhar)
India, 2004

REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE
OF

MASTERS OF SCIENCE IN INTERNETWORKING

(The Department of Computing Science)



THE UNIVERSITY OF ALBERTA
August, 2007



DEDICATED To

My Family...

“Mummy”, “Papa”, “Kimmy” & “Gudia”

*A*BSTRACT

*T*raffic shaping is a way to help increase network performance by controlling the amount of data that flows into and out of the network. Traffic is categorized, queued, and directed according to network policies. Intelligent shaping schemes can guarantee a particular Quality of Service (often measured in jitter, packet loss, and latency) for an application or a user while still allowing other traffic to use all remaining bandwidth.

Apparently, one of the most expanding areas of research in computing science, traffic engineering, aims to balance and optimize the service agreements and the cost of network operation. This stands up to be the most business critical concerns in today's market. Every organization has a different set of requirements depending upon a list of factors that are specific to its network infrastructure. Thus, the choice of the shaping mechanism used also change accordingly. Additionally, the behavior of each shaping mechanism depends on the queuing type used with it. So, one of the challenges in this context would be to chose the right combination that fits best with the organization's needs.

Our major concerns in this project will include controlling access to available bandwidth, ensuring that traffic conforms to the policies established for it, and regulating the flow of traffic in order to avoid congestion that can occur when the sent traffic exceeds the access speed of its remote, target interface.

Keeping these concerns in reference, we will study the concepts behind various traffic shaping techniques and will also try to test each of these techniques in a simulated real-time environment. We have setup a very proficient network model that mimics a real-time scenario of a medium-business organization. All the demonstrations are done using this network model.

CONTENTS

<i>A</i> BSTRACT.....	iii
<i>C</i> ONTENTS.....	iv
<i>L</i> IST OF <i>F</i> IGURES.....	ix
<i>L</i> IST OF <i>T</i> ABLES.....	xi
<i>A</i> CKNOWLEDGEMENTS.....	xii
<i>C</i> HAPTER 1 – The Preface.....	1
1.1 Background.....	2
1.2 Traffic Shaping.....	2
1.2.1 Types of Traffic Shaping.....	2
1.2.2 Ways of Managing Bandwidth with Traffic Shaping.....	4
1.2.3 Issues and Challenges.....	5
1.3 Contribution.....	7
1.4 Report Organization.....	7
1.5 Chapter 2 – Preview.....	8
<i>C</i> HAPTER 2 – Traffic Regulation Mechanisms.....	9
2.1 Why do we need traffic regulation mechanisms?	9
2.2 Traffic Regulation Mechanisms.....	10
2.2.1 The Token Bucket.....	11
2.2.2 Policing vs. Shaping.....	14

2.3 Traffic Policing.....	17
2.3.1 Benefits.....	17
2.3.2 Restrictions.....	18
2.4 Traffic Shaping.....	18
2.4.1 Traffic Shaping and Rate of Transfer.....	18
2.4.2 Traffic Shaping Parameters.....	19
2.4.3 Traffic Shaping and Queuing.....	20
2.5 Chapter 3 – Preview.....	20
CHAPTER 3 – The Infrastructure.....	21
3.1 Scoop.....	21
3.2 The Blueprint.....	22
3.2.1 Logical topology.....	23
3.2.3 Physical topology.....	24
3.3 The Device Catalogue.....	25
3.4 The Big Picture.....	28
3.5 IP addressing.....	30
3.6 Protocols used.....	32
3.6.1 Ethernet.....	32
3.6.2 Frame Relay.....	32
3.6.3 Border Gateway Protocol (BGP).....	33
[A] BGP Attributes.....	34
[B] BGP Path Selection.....	40
[C] Summarization.....	40
3.6.4 Open Shortest Path First (OSPF).....	40
3.6.5 TCP and UDP.....	42
3.6.6 Real Time Protocol (RTP).....	42
3.6.7 Network Time Protocol (NTP).....	43
3.6.8 Skinny Client Protocol (SCCP).....	43
3.7 The Toolkit.....	44
3.7.1 Monitoring Tools	44
[A] Ethereal ver0.10.14.....	44
[B] VLC Player ver3.4.5.....	44
[C] Router IOS Debug commands.....	44
3.7.2 Cisco Call Manager Express (CME).....	44
[A] Setting up DHCP service for Cisco CME.....	45
[B] Setting IP phones with CME.....	45

[C] Setting up initial extensions and Phones.....	47
3.8 Chapter 4 – Preview.....	48
CHAPTER 4 – Queuing Theories.....	49
4.1 Definition.....	49
4.2 First-In-First-Out Queuing.....	50
4.2.1 Features	51
4.2.2 Limitations.....	51
4.2.3 Configuring FIFO.....	51
4.3 Priority Queuing.....	52
4.3.1 Features	53
4.3.2 Limitations.....	53
4.3.3 Configuring PQ.....	54
4.4 Custom Queuing.....	54
4.4.1 Features.....	55
4.4.2 Limitations.....	55
4.4.3 Configuring CQ.....	56
4.5 Weighted Fair Queuing.....	56
4.5.1 Features	57
4.5.2 Limitations.....	58
4.5.3 Configuring WFQ.....	58
4.6 IP RTP Prioritization.....	59
4.6.1 Features	59
4.6.2 Limitations.....	60
4.6.3 Configuring IP RTP Prioritization.....	60
4.7 Chapter 5 – Preview.....	60
CHAPTER 5 – Generic Traffic Shaping.....	61
5.1 The Concept.....	61
5.1.1 Queuing.....	62
5.1.2 GTS Features.....	63
5.1.3 Restrictions.....	63
5.2 Configuring GTS.....	64
5.2.1 Configuring GTS on an interface.....	64
5.2.2 Configuring GTS using an Access Control List.....	64

5.2.3 Configuring Adaptive GTS for Frame Relay networks	65
5.3 Simulation Study.....	66
5.3.1 Experimental Setup.....	66
5.3.2 Results Analysis.....	68
5.3.3 Conclusion.....	72
5.4 Chapter 6 – Preview.....	73
CHAPTER 6 – Class based Traffic Shaping.....	74
6.1 The Concept.....	74
5.1.1 Queuing.....	75
5.1.2 CBTS Features.....	75
5.1.3 Restrictions.....	76
6.2 Configuring CBTS.....	77
6.2.1 CBTS in a Primary level (Parent) Policy Map.....	77
6.2.2 CBTS with Secondary level (Child) Policy Map.....	77
6.3 Simulation Study.....	78
6.3.1 Experimental Setup.....	78
6.3.2 Results Analysis.....	81
6.3.3 Conclusion.....	83
6.4 Chapter 7 – Preview.....	83
CHAPTER 7 – Frame Relay Traffic Shaping.....	84
7.1 The Concept.....	84
7.1.1 Queuing.....	87
7.1.2 FRTS Features.....	88
7.1.3 Restrictions.....	88
7.2 Configuring FRTS.....	89
7.3 Simulation Study.....	90
6.3.1 Experimental Setup.....	90
6.3.2 Results Analysis.....	94
6.3.3 Conclusion.....	97

A P P E N D I X - *I* – Device Configuration Files..... 98

A P P E N D I X - *II* – Project Life Cycle..... 118

B I B L I O G R A P H Y..... 119

.....

LIST OF FIGURES

Chapter - 2

2.1 Scenario: Variable speed access links over the communication path.....	9
2.2 Scenario: Limited access to resources.....	10
2.3 Scenario: TDM or leased line is implemented over a single physical link.....	10
2.4 Traffic Shaping Classifier, Meter and Marker.....	11
2.5 Graph illustrating exceeding and conforming traffic.....	11
2.6 Token Bucket Conform action.....	13
2.7 Token Bucket Exceed action.....	13
2.8 $T_c = B_c / CIR$	14
2.9 Shaping vs Policing (Graphical representation).....	15
2.10 Relationship between various Traffic Shaping Parameters.....	19

Chapter – 3

3.1 Network Model: Logical Topology.....	23
3.2 Network Model: Physical Topology.....	24
3.3 Finite State Machine: BGP events.....	34
3.4 Local Preference Attribute.....	35
3.5 Multi Exit Discriminator (MED) attribute.....	37
3.6 AS_Path Attribute.....	39
3.7 OSPF Areas.....	41
3.8 Real Time Protocol (RTP).....	42

Chapter – 4

4.1 Queuing actions occurring before transmission.....	49
4.2 First-In-First-Out Queuing.....	50
4.3 Priority Queuing.....	52
4.4 Four Types of queues within Priority Queuing.....	53
4.5 Custom Queuing.....	55
4.6 Weighted Fair Queuing.....	57
4.7 IP RTP Prioritizaion.....	59

Chapter -5

5.1 Generic Traffic Shaping Mechanism.....	61
--------------------------------------------	----

5.2 Queuing in GTS.....	62
5.3 GTS Experimental Setup.....	66
5.4 Captures from Ethereal.....	72
5.5 Captures from Ethereal.....	73

Chapter – 6

6.1 CBTS Experimental setup.....	79
6.2 CBTS Hierarchical Policy Map Structure.....	80

Chapter – 7

7.1 Frame Relay Traffic Shaping Mechanism.....	85
7.2 FRTS process flow chart.....	86
7.3 Priority Queuing for FRTS.....	87
7.4 FRTS Experimental Setup.....	91

A P P E N D I X – II

AII-1 Project Development Life Cycle.....	118
-------------------------------------------	-----

.....

LIST OF TABLES

1-1 Basic differences between the various Traffic Shaping Techniques.....	3
2-1 Differences between shaping and policing.....	15
3-1 The icons used in this report.....	22
3-2 Functions of the network components used in the Model.....	25
3-3 IP Addressing Scheme.....	30

.....

ACKNOWLEDGEMENT

This document is the consolidated report of the project “Traffic Shaping Networks”. Its accomplishment is highly indebted to the invaluable contribution from an assortment of personalities. Here, I would like to express my gratitude to all those who gave me the possibility to complete this report.

The project would not have been possible without the administrative and technical will of Dr. M.H. (Mike) MacGregor, MINT Program Director at the University of Alberta. I am deeply indebted to him as well as his stimulating suggestions and encouragements that helped me in all the time of the project and writing of this report. I would also want to thank the Department of Computing Science of the University of Alberta, for giving me permission to commence this project in the first instance, to do the necessary experimentation work and to use departmental resources and equipments.

Next, I would like to thank Mr. Cesar Barrero (CCIE # 6994), Systems Engineer, Cisco Systems, Edmonton, for his timely support and advice. I would also like to express my gratefulness to my colleague, Gurvir Gill, for being an incredible partner in the project. It was a pleasure working with him.

Above all, I pay my deepest gratitude to the Almighty for blessing me with all the potential and the capabilities to be able to establish my work. Lastly, I truly believe that I cannot thank enough to my family without whom my existence is meaningless. Their love and support has always been and will always be my greatest strengths and a driving force to accomplish my ambition.

.....

CHAPTER 1

The Preface

1.1 Background

In the past few decades, the advances in computing science and engineering, and technological improvements in network and communication infrastructure have revolutionized the computing and communication environment around us. The eighties and nineties of the past centuries have seen a rapid growth of the World Wide Web that has an immensely significant impact in our day-to-day lives. Additionally, wide-spread use of mobile devices like lap-tops, palm-tops, PDAs, mobile phones, etc is redefining the way in which the information is exchanged between such devices in different parts of the geographic region. Integration of local and global information provided by thousands or even millions of such devices has started to make an impact on the information processing and the protocols involved in bandwidth usage...

Most enterprises face increasing network capacity and performance challenges as they roll out enterprise resource planning (ERP), voice over IP (VOIP), storage over IP and other sophisticated applications across their wide area networks. Traffic shaping and quality of service (QOS) tools, thus prove to be very important weapons for optimizing the performance of those applications because they allow IT managers to guarantee the amount of bandwidth available to mission-critical processes.

When most IT departments consider the prospect of seizing granular control over user and application traffic, they're optimistic about solving their performance problems once and for all. Unfortunately, traffic shaping is a zero-sum game: The bandwidth reserved for one application must be taken away from the bandwidth available to another application. Like it or not, IT managers doing traffic shaping must play favorites, deciding which applications are more important and making some users happier while reducing performance for others. In enterprises with remote facilities connected via low-bandwidth links, such traffic shaping decisions can be particularly painful.

But it doesn't have to be that way. It's a lot easier to divvy up traffic if there's enough bandwidth to easily accommodate everything in the first place, for example. That's why some enterprises use compression to increase the bandwidth pie before using traffic shaping to slice it up.

Nevertheless, other issues such as link latency can slow application performance even if we're using QOS and compression. But there are lots of solutions to performance problems, and working together, they can have a far greater impact on WAN link and application performance than QOS alone, allowing us to improve performance without turning some users or applications into losers.

To get the maximum benefit from traffic shaping, compression, or other network performance technologies, it pays to take a holistic approach to the problem. Rather than simply noticing one application whose performance can be improved, it is required to look at the entire network system, at all the applications, and at the link speeds to the company's remote sites; using network and traffic monitoring tools to pinpoint the biggest trouble spots by location and application, and then evaluating compression and QOS/Shaping products on a cost/benefit scale.

If bandwidth contention is the biggest problem, that would point to a QOS/Traffic-Shaping oriented solution. However if capacity is the main problem, then compression is the solution. Finally, interactions if protocol or sheer distance from point to point is the issue, then point towards a TCP optimization product. With the right mix, we can avoid diminishing anyone's performance without breaking the bank.

This report explores the traffic control schemes for multiple access, non-broadcast media such as Frame Relay and ATM, where the physical access speed varies between two endpoints. Traffic shaping technology is used to accommodate mismatched access speeds. It also includes, study of other commonly used traffic shaping techniques as well.

1.2 Traffic Shaping

Definition: Traffic Shaping is a mechanism that alters the traffic characteristics of a stream of packets on a connection to achieve better network efficiency, while meeting the QoS objectives, or to ensure conformance at a subsequent interface. Traffic shaping must maintain packet sequence integrity on a connection. Shaping modifies traffic characteristics of a packet flow with the consequence of increasing the mean delay.

Traffic shaping causes the traffic to be categorized, queued, and directed according to network policies. Intelligent shaping schemes can guarantee a particular Quality of Service (often measured in jitter, packet loss, and latency) for an application or a user while still allowing other traffic to use all remaining bandwidth.

1.2.1 Types of Traffic Shaping

Broadly speaking, Cisco IOS supports four types of traffic shaping. These are similar in implementation, sharing the same code and data structures, but they differ in regard to their CLIs and the queue types they use.

- Generic Traffic Shaping: GTS shapes traffic by reducing outbound traffic flow to avoid congestion by constraining traffic to a particular bit rate using the token bucket mechanism
- Class-Based Shaping: Peak and average traffic shaping is configured on a per-interface or per-class basis
- Distributed Traffic Shaping: The DTS feature provides a method of managing the bandwidth of an interface to avoid congestion, to meet remote site requirements, and to conform to a service rate that is provided on that interface
- Frame Relay Traffic Shaping: FRTS can eliminate bottlenecks in Frame Relay networks that have high-speed connections at the central site and low-speed connections at branch sites. We can configure rate enforcement—a peak rate configured to limit outbound traffic—to limit the rate at which data is sent on the VC at the central site.

Here are a few ways in which these mechanisms differ:

- For GTS, the shaping queue is a weighted fair queue. For FRTS, the queue can be a weighted fair queue (configured by the frame-relay fair-queue command), a strict priority queue with WFQ (configured by the frame-relay ip rtp priority command in addition to the frame-relay fair-queue command), custom queueing (CQ), priority queueing (PQ), or FIFO.
- For Class-Based Shaping, GTS can be configured on a class, rather than only on an access control list (ACL). In order to do so, we must first define traffic classes based on match criteria including protocols, ACLs, and input interfaces. We can then apply traffic shaping to each defined class.
- FRTS supports shaping on a per-DLCI basis; GTS and DTS are configurable per interface or subinterface.
- DTS supports traffic shaping based on a variety of match criteria, including user-defined classes, and DSCP.

Table 1.1: Basic differences between the various Traffic Shaping Techniques:

Mechanism	GTS	Class-Based	DTS	FRTS
Command-Line Interface	<ul style="list-style-type: none"> • Applies parameters per subinterface • traffic group command supported 	<ul style="list-style-type: none"> • Applies parameters per interface or per class 	<ul style="list-style-type: none"> • Applies parameters per interface or subinterface 	<ul style="list-style-type: none"> • Classes of parameters • Applies parameters to all virtual circuits (VCs) on an interface through inheritance mechanism • No traffic group command
Queues Supported	<ul style="list-style-type: none"> • WFQ per subinterface 	<ul style="list-style-type: none"> • CBWFQ inside GTS 	<ul style="list-style-type: none"> • WFQ, strict priority queue with WFQ, CQ, PQ, first-come, first-served (FCFS) per VC 	<ul style="list-style-type: none"> • WFQ, strict priority queue with WFQ, CQ, PQ, FCFS per VC

1.2.2 Ways of Managing Bandwidth with Traffic Shaping

Wide area (Internet) bandwidth can be actively managed to limit traffic in several different ways, using traffic shaping techniques:

1. Per-application rules. Traffic shapers can identify and categorize specific types of network traffic, constraining each particular category of traffic to use no more than a specified amount of bandwidth. For example, we might hypothetically have a rule that limits aggregate FTP traffic to no more than 6 megabits per second and another rule that limits total streaming audio traffic to no more than 3 Mbps, etc.

Traffic shapers can categorize traffic based on macroscopic characteristics, such as the traffic's protocol (IP, IPX, AppleTalk, DECNet, etc.), the ports an application is known to use (for example, Kazaa typically runs on port 1214), or on the basis of connections to a particular well-known host (such as a central game server), etc.

Traffic can also be categorized based on the content of the flow regardless of the flow's macroscopic characteristics. For example, most traffic shapers can easily identify and automatically categorize web traffic based on the negotiations that take place between a web server and a web browser when a page is requested, regardless of whether the web server is running on port 80 (the default) or some other nonstandard port.

2. Per-user rules. Traffic shapers can set per-user traffic limits to ensure that network traffic is shared fairly among all users. For instance, we might decide to use a per-user rule that limits traffic to or from each user to no more than 256Kbps (giving them DSL-like service). When traffic is limited in that way, a user can still access whatever he or she wants, but the flows are "smoothed out" to a specified level rather than attempting to use all or much of the total available network capacity campuswide.

Traffic limits can be either "hard" or "burstable." As we might expect, a hard limit is a fixed ceiling that can't be exceeded. Burstable limits, on the other hand, allow traffic to exceed the base threshold value (at least up to a specified "burst limit") as long as capacity remains available and there's no higher priority application preemptively claiming that capacity.

3. Priority management. In addition to setting hard or burstable traffic limits on a per-application or per-user basis, traffic shaping devices can also be used to define the relative importance, or priority, of different types of traffic. For example, in an academic network where teaching and research are most important, recreational uses of the network (such as network games or peer-to-peer file sharing application traffic) can be allowed bandwidth only when higher priority applications don't need it.

Some traffic shaping tasks can be done directly on a regular Cisco router, just as a router can also be used to do some firewall-like packet filtering tasks. However, specialized traffic shapers, like any specialized devices, can be optimized to

specifically and efficiently handle their unique responsibilities. Specialized devices also typically have a "bigger bag of tricks" to draw from when dealing with problems in their special area of expertise. Doing traffic shaping on a dedicated traffic shaping box also avoids loading up routers with other tasks, leaving the router free to focus on doing its job of routing packets as fast as it can.

1.2.3 Issues and Challenges

There are a number of technical issues and challenges associated with the technique of traffic shaping. The three most significant of those are:

- *How to Increase Available Bandwidth?*

There are several approaches to solving a problem of insufficient bandwidth:

 - The best approach is to increase the link capacity to accommodate all applications and users with some extra bandwidth to spare. This solution sounds simple enough but in the real world it brings a high cost in terms of the money and time it takes to implement.
 - Another option is to classify traffic into QoS classes and prioritize it according to importance (business-critical traffic should get enough bandwidth, voice should get enough bandwidth and prioritized forwarding and the least important traffic should get the remaining bandwidth). There are a wide variety of mechanisms available in the Cisco IOS that provide bandwidth guarantees, for example:
 - Priority or Custom Queuing
 - Modified Deficit Round Robin (on Cisco 12000 series routers)
 - Distributed ToS-based and QoS-group-based Weighted Fair Queuing (on Cisco 7x00 series routers)
 - Class-based Weighted Fair Queuing
 - Optimizing link usage by compressing the payload of frames (virtually) increases the link bandwidth. Compression, on the other hand, also increases delay due to complexity of compression algorithms. Using hardware compression can accelerate the compression of packet payloads. Stacker and Predictor are two compression algorithms available in Cisco IOS. Another link efficiency mechanism is header compression. This mechanism is especially effective in networks where most packets carry small amounts of data (payload-to-header ratio is small). Typical examples of header compression are TCP Header Compression and RTP Header Compression.
- *How to Reduce Delay?*

Assuming that a router is powerful enough to make a forwarding decision in a negligible time it can be said that most of the processing, queuing delay and propagation delay is influenced by the following factors:

- Average length of the queue
- Average length of packets in the queue
- Link bandwidth

By minimizing delay, jitter is also reduced (delay is more predictable). There are several approaches to accelerate packet dispatching of delay-sensitive flows:

- Increase link capacity. Enough bandwidth causes queues to shrink, making sure packets do not have to wait long before they can be transmitted. Additionally, more bandwidth reduces serialization time. On the other hand, this might be an unrealistic approach due to the costs associated with the upgrade.
 - A more cost-effective approach is to enable a queuing mechanism that can give priority to delay-sensitive packets by forwarding them ahead of other packets. There are a wide variety of queuing mechanisms available in Cisco IOS that have pre-emptive queuing capabilities, for example: Priority Queuing, Custom Queuing, Strict-priority or Alternate Priority queuing within the Modified Deficit, Round Robin (on Cisco 12000 series routers), IP RTP prioritization, Class-based Low-latency Queuing
 - Payload compression reduces the size of packets and, therefore, virtually increases link bandwidth. Additionally, compressed packets are smaller and need less time to be transmitted. On the other hand, compression uses complex algorithms that take time and add to the delay. This approach is, therefore, not used to provide low-delay propagation of packets.
 - Header compression on the other hand is not as CPU-intensive and can be used in combination with other mechanisms to reduce delay. It is especially useful for voice packets that have a bad payload-to-header ratio, which is improved by reducing the header of the packet (RTP header compression).
- How to prevent packet loss?
 Packet loss is usually a result of congestion on an interface. Most applications that use TCP experience slow down due to TCP adjusting to the network's resources (dropped TCP segments cause TCP sessions to reduce their window sizes). There are some other applications that do not use TCP and cannot handle drops (fragile flows). The following approaches can be taken to prevent drops of sensitive applications:
 - Increased link capacity to ease or prevent congestion.
 - Guarantee enough bandwidth and increase buffer space to accommodate bursts of fragile applications. There are several mechanisms available in Cisco IOS that can guarantee bandwidth and/or provide prioritized forwarding to drop sensitive applications, for example: Priority Queuing, Custom Queuing, Modified Deficit Round Robin (on Cisco 12000 series routers), IP RTP prioritization, Class-based Weighted Fair Queuing, Class-based Low-latency Queuing
 - Prevent congestion by dropping other packets before congestion occurs. Weighted Random Early Detection can be used to start dropping other packets

before congestion occurs. There are some other mechanisms that can also be used to prevent congestion:

1. Traffic Shaping delays packets instead of dropping them (Generic Traffic Shaping, Frame Relay Traffic Shaping and Class-based Shaping).
2. Traffic Policing can limit the rate of less important packets to provide better service to drop-sensitive packets (Committed Access Rate and Class-based Policing).

1.3 Contribution

*I*n this project we have made an effort to consummate a comprehensive analysis of the techniques used to alter the traffic characteristics of packet streams. This is done in order to achieve better efficiency in terms of network performance. This technique, formally known as Traffic Shaping, has numerous applications depending on the business needs. There is a commendable amount of flexibility in these techniques that allow users to expect customized results.

We have created a lab setup simulating a real-time Client-to-ISP environment, which is discussed in detail in chapter 3. The client is a medium-business company having offices in Edmonton and Calgary. The concern is to make effective usage of the available bandwidth so as to ensure reliable QoS for their voice and video data streams.

We have implemented and tested the various Shaping techniques on this model by devising different scenarios and possibilities, and then choosing the best-fit option out of the available shaping and queuing methods. The results and the changes are captured using packet sniffers and the router debuggers.

1.4 Report Organization

*T*his report starts with a brief discussion about the various bandwidth issues prevalent in real time networks, in Chapter 1. We also talk about the various factors causing bandwidth starving and the ways of managing it. The fundamentals of the two basic Traffic Regulation Mechanisms are discussed in Chapter 2. The underlying protocol used in these mechanisms, Token Bucket, is also discussed in detail.

Next, in Chapter 3 we discuss the Lab setup used by us in testing and demonstrating the various traffic shaping techniques. This chapter covers detailed information about the network model, devices, testing tools and protocols used in accomplishing the objective of the report.

Chapter 4, is a brief overview of the various queuing mechanisms. This discussion was needed in order to have a commendable understanding of the behavior of the queues in the corresponding traffic shaping mechanisms discussed in the next section. The next three Chapters (5, 6 and 7) constitute a detailed explanation of the conceptual and the practical know-how of three of the most prominent shaping mechanisms viz. – Generic Traffic Shaping, Class Based Traffic Shaping and Frame Relay Traffic Shaping, respectively. In each case we define a Lab scenario followed by the implementation and testing of the corresponding technique.

The report ends with two Appendices - Appendix-I consists of all the configuration files for all the networking devices used in the network model and Appendix-II includes a diagrammatic representation of the Life Cycle of the report.

1.5 Chapter 2 - *Preview*

*T*raffic shaping is used to regulate a stream of packets on a connection to achieve better network efficiency. Traffic Policing on the other hand is used to accomplish the same objective but with a bit of variance. The next chapter discusses in brief, the main concepts of these two prominent traffic shaping mechanisms and also lists the differences between the two of these. Chapter 2, also explains Token Bucket Algorithm, which is the underlying principle used by both of these traffic shaping mechanism

.....

CHAPTER 2

Traffic Regulation Mechanisms

2.1 Why do we need Traffic Regulation Mechanisms?

Traffic Regulation / Rate limiting is typically used to satisfy one of the following requirements:

- Prevent and manage congestion in ATM and Frame Relay networks, where asymmetric bandwidths are used along the traffic path. This prevents the layer-2 network from dropping large amounts of traffic by differentially dropping excess traffic at ingress to the ATM or Frame Relay networks based on Layer-3 information (for example: IP precedence, DSCP, access list, protocol type, etc.) Limit the access rate on an interface when high-speed physical infrastructure is used in transport, but sub-rate access is desired.
- Engineer bandwidth so that traffic rates to certain applications or classes of traffic follow a specified traffic -rate policy.
- Implement a virtual TDM system, where an IP network is used, but has the bandwidth characteristics of a TDM system (that is, fixed maximum available bandwidth). Inbound and outbound policing can, for example, be used on one router to split a single point-to-point link into two or more virtual point-to-point links by assigning a portion of the bandwidth to each class, thus preventing any class from monopolizing the link in either direction.

These traffic regulation mechanisms have their applications in a number of network scenarios. Some of the most common examples ones are:

Example1: Variable access links over the communication path. (Figure 2.1)

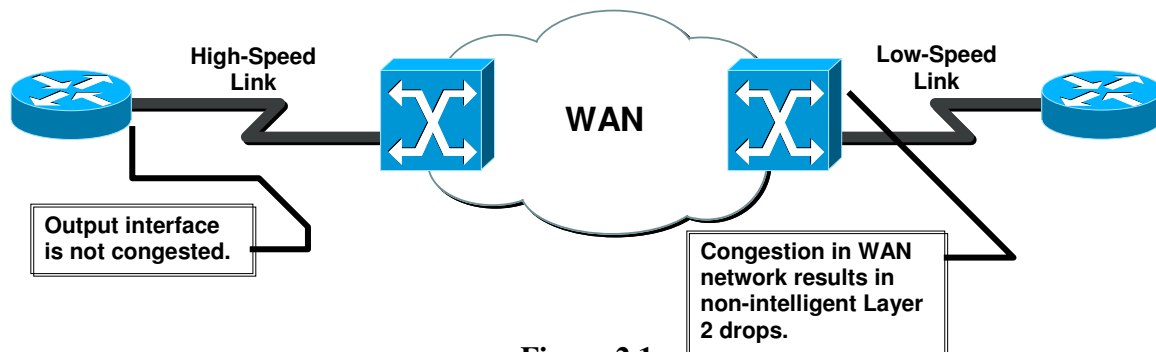


Figure 2.1

Example 2: Access to resources is limited. (Figure 2.2)

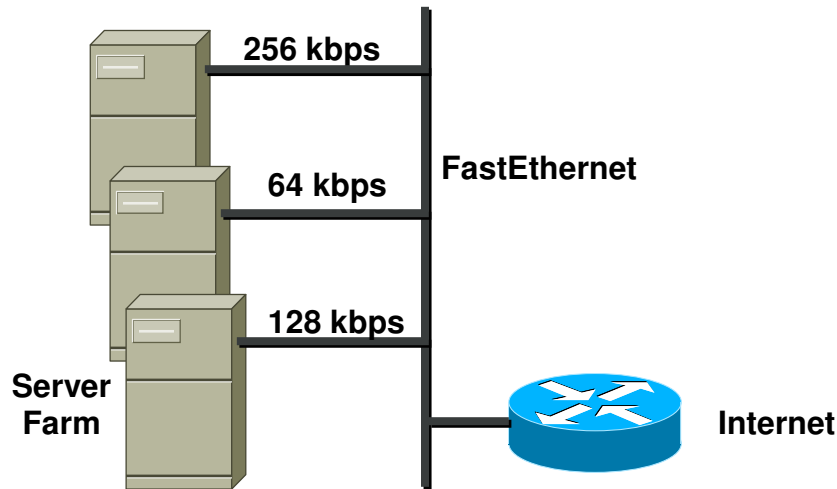


Figure 2.2

Example 3: A virtual TDM or leased line is implemented over a single physical link on one side (Figure 2.3)

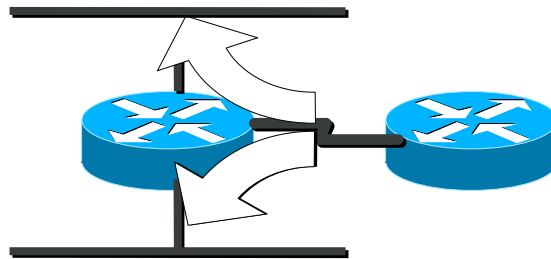


Figure 2.3

In this chapter we give a brief description of the Cisco IOS QoS traffic policing and shaping mechanisms.

2.2 Traffic Regulation Mechanisms

The Cisco IOS QoS offers two kinds of traffic regulation mechanisms—POLICING AND SHAPING.

The rate-limiting features of committed access rate (CAR) and the Traffic Policing feature provide the functionality for policing traffic. The features of Generic Traffic Shaping (GTS), Class-Based Shaping, Distributed Traffic Shaping (DTS), and Frame Relay Traffic Shaping (FRTS) provide the Functionality for shaping traffic.

Traffic shaping and policing mechanisms are used to rate-limit traffic classes. They have to be able to classify packets and meter their rate of arrival.

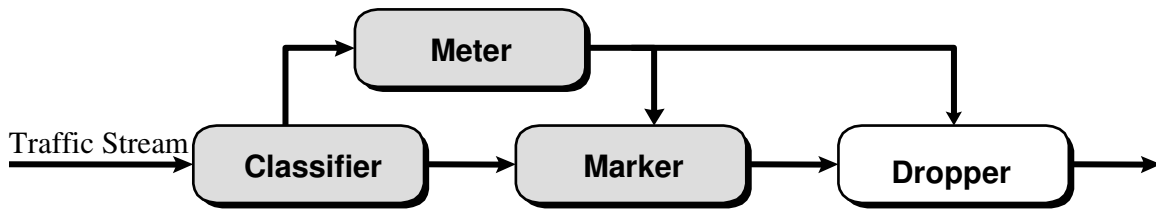


Figure 2.4

Policers and shapers usually identify traffic descriptor violations in an identical manner. They usually differ, however, in the way they respond to violations, for example:

A policer typically drops traffic. (For example, the CAR rate-limiting policer will either drop the packet or rewrite its IP precedence, resetting the type of service bits in the packet header.)

A shaper typically delays excess traffic using a buffer, or queueing mechanism, to hold packets and shape the flow when the data rate of the source is higher than expected. (For example, GTS and Class-Based Shaping use a weighted fair queue to delay packets in order to shape the flow, and DTS and FRTS use either a priority queue, a custom queue, or a FIFO queue for the same, depending on how we configure it.)

2.2.1 The Token Bucket

In order to measure the traffic rate; routers use the token bucket mathematical model. It keeps track of packet arrival rate. The token bucket model is used whenever a new packet is processed and the return value is either “conform” or “exceed” as shown in figure 2.5 in the following page.

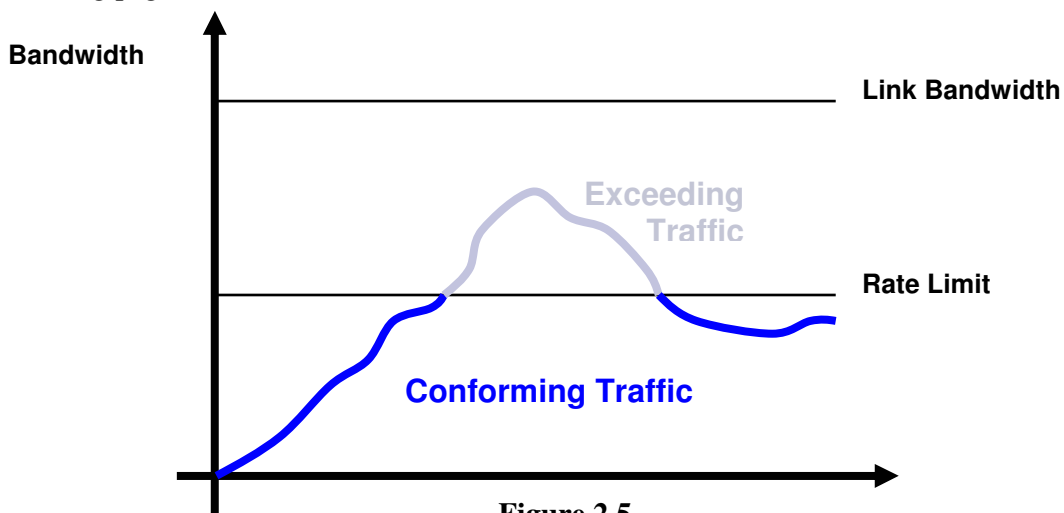


Figure 2.5

Since both policing and shaping use the token bucket mechanism, lets explains how a token bucket works.

A token bucket is a formal definition of a rate of transfer. It has three components: a burst size, a mean rate, and a time interval (T_c). Although the mean rate is generally represented as bits per second, any two values may be derived from the third by the relation shown as follows:

$$\text{Mean Rate} = \text{Burst Size} / \text{Time Interval.}$$

Here are some definitions of these terms:

- *Mean rate*—Also called the committed information rate (CIR), it specifies how much data can be sent or forwarded per unit time on average.
- *Burst size*—Also called the Committed Burst (B_c) size, it specifies in bits (or bytes) per burst how much traffic can be sent within a given unit of time to not create scheduling concerns. (For a shaper, such as GTS, it specifies bits per burst; for a policer, such as CAR, it specifies bytes per burst.)
- *Time interval*—Also called the measurement interval, it specifies the time quantum in seconds per burst.

By definition, over any integral multiple of the interval, the bit rate of the interface will not exceed the mean rate. The bit rate, however, may be arbitrarily fast within the interval. A token bucket is used to manage a device that regulates the data in a flow. For example, the regulator might be a traffic policer, such as CAR, or a traffic shaper, such as FRTS or GTS. A token bucket itself has no discard or priority policy. Rather, a token bucket discards tokens and leaves to the flow the problem of managing its transmission queue if the flow overdrives the regulator. (Neither CAR nor FRTS and GTS implement either a true token bucket or true leaky bucket.)

In the token bucket metaphor, tokens are put into the bucket at a certain rate. The bucket itself has a specified capacity. If the bucket fills to capacity, newly arriving tokens are discarded. Each token is permission for the source to send a certain number of bits into the network. To send a packet, the regulator must remove from the bucket a number of tokens equal in representation to the packet size. If the bucket fills to capacity, newly arriving tokens are discarded. Discarded tokens are not available to future packets. If there are not enough tokens in the bucket to send the packet, the regulator may:

- Wait for enough tokens to accumulate in the bucket (traffic shaping)
- Discard the packet (policing)

The figure 2.6 shows a token bucket, with the current capacity of 700 bytes. When a 500-byte packet arrives at the interface, its size is compared to the bucket capacity (in bytes). The packet conforms to the rate limit (500 bytes < 700 bytes), and the packet is forwarded. 500 tokens are taken out of the token bucket leaving 200 tokens for the next packet.

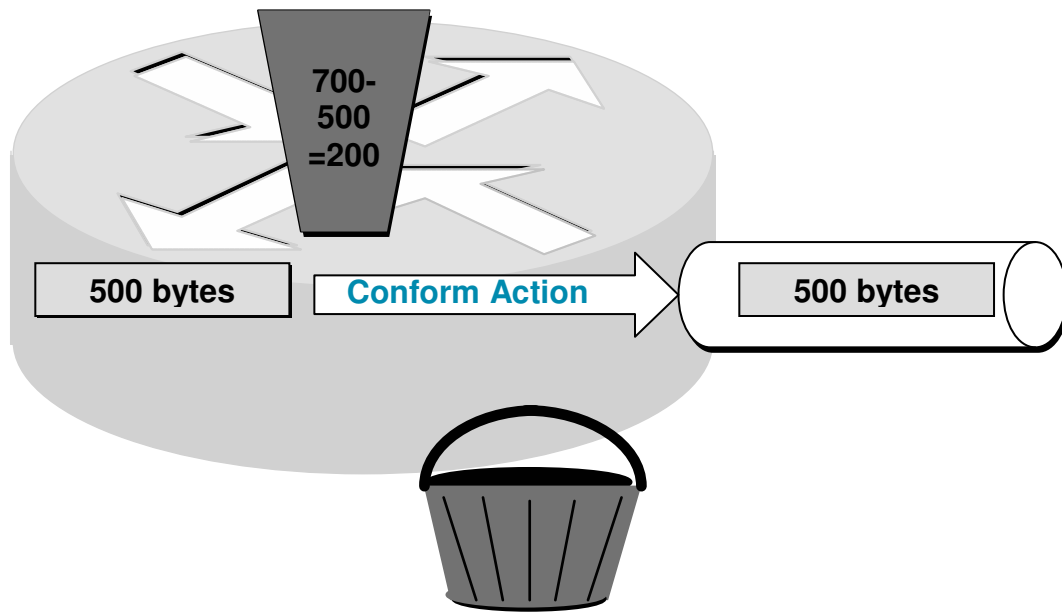


Figure 2.6

However, when the next packet arrives immediately after the first packet, and no new tokens have been added to the bucket (which is done periodically), the packet exceeds the rate limit. The packet size is greater than the current capacity of the bucket, and the exceed action is performed (“drop” in the case of pure policing, “delay” in the case of shaping).

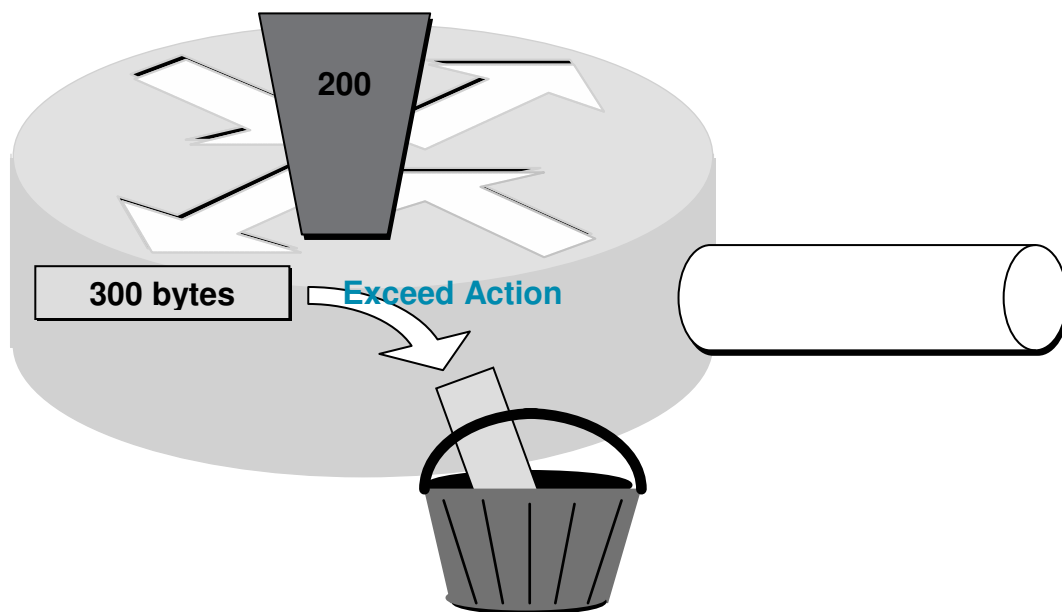


Figure 2.7

Token bucket implementations usually rely on three parameters: CIR, B_c and B_e . CIR is the Committed Information Rate (also called the committed rate, or the shaped rate). B_c is known as the burst capacity. B_e is known as the excess burst capacity. T_c is an interval constant that represents time. A B_c of tokens are forwarded without constraint in every T_c interval.

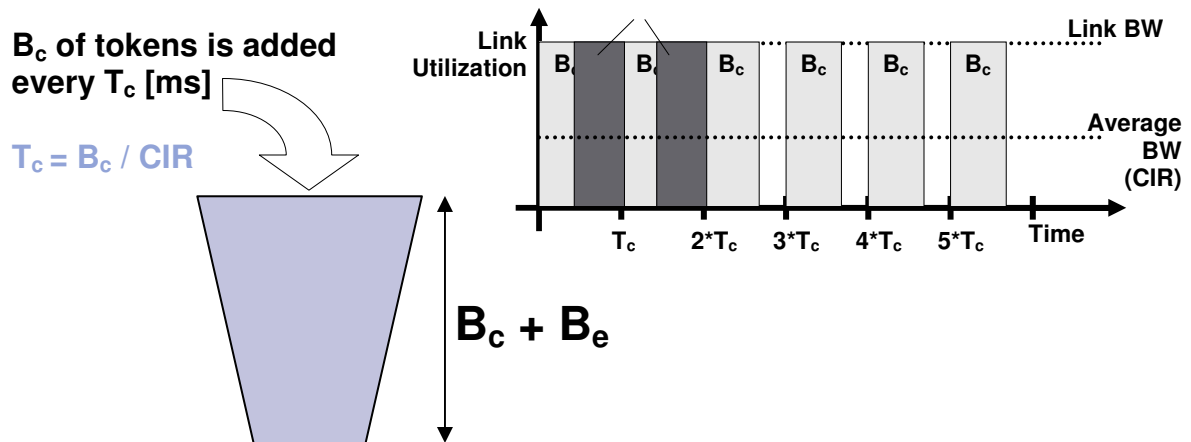


Figure 2.8

In the token bucket metaphor, tokens are put into the bucket at a certain rate, which is B_c tokens every T_c seconds. The bucket itself has a specified capacity. If the bucket fills to capacity ($B_c + B_e$), it will overflow and therefore newly arriving tokens are discarded. Each token grants permission for a source to send a certain number of bits into the network. To send a packet, the regulator must remove, from the bucket, the number of tokens equal in representation to the packet size.

For example, if 8000 bytes worth of tokens are placed in the bucket every 125 milliseconds, the router can steadily transmit 8000 bytes every 125 milliseconds, if traffic constantly arrives at the router.

If there is no traffic at all, 8000 bytes per 125 milliseconds get accumulated in the bucket, up to the maximum size ($B_c + B_e$). One second's accumulation therefore collects 64000 bytes worth of tokens, which can be transmitted immediately in the case of a burst. The upper limit, $B_c + B_e$, defines the maximum amount of data, which can be transmitted in a single burst, at the line rate.

2.2.2 Policing vs. Shaping.

The key difference between Policing and Shaping is that Traffic policing propagates bursts whereas Traffic shaping tends to soothe the bursts by using queuing mechanisms. When the

traffic rate reaches the configured maximum rate, excess traffic is dropped (or remarked). The result is an output rate that appears as a saw-tooth with crests and troughs. In contrast to policing, traffic shaping retains excess packets in a queue and then schedules the excess for later transmission over increments of time. The result of traffic shaping is a smoothed packet output rate (see figure 2.9).

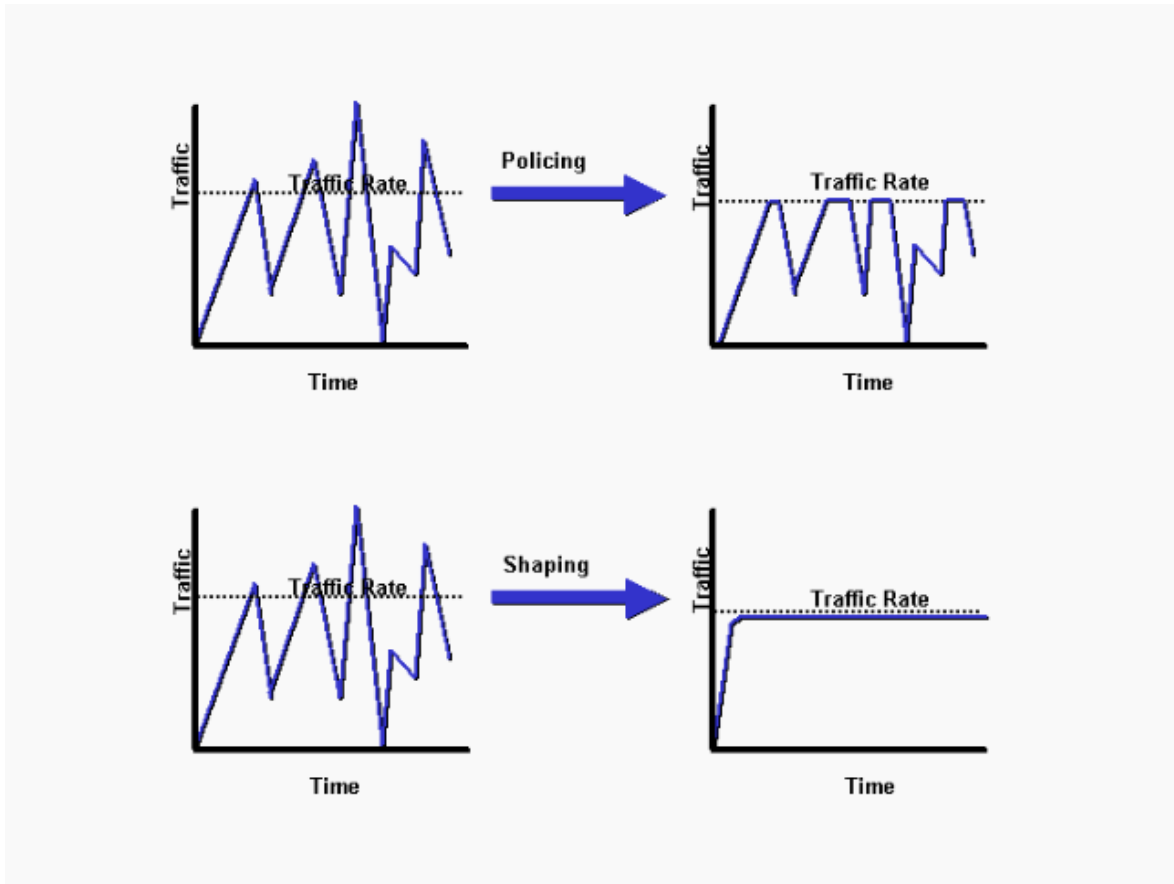


Figure 2.9

Table 2.1: The following table lists the differences between shaping and policing to help we choose the best solution.

	Shaping	Policing
Objective	Buffer and queue excess packets above the committed rates.	Drop (or remark) excess packets above the committed rates. Does not buffer.
Token Refresh Rate	Incremented at the start of a time interval. (Minimum number of intervals is required.)	Continuous based on formula: $1 / \text{committed information rate}$
Token Values	Configured in bits per second.	Configured in bytes.

Configuration Options	Shape command in the modular quality of service command-line interface (MQC) to implement class-based shaping. frame-relay traffic-shape command to implement Frame Relay Traffic Shaping (FRTS). traffic-shape command to implement Generic Traffic Shaping (GTS).	police command in the MQC to implement class-based policing. rate-limit command to implement committed access rate (CAR).
Applicable on Inbound	No	Yes
Applicable on Outbound	Yes	Yes
Bursts	Controls bursts by smoothing the output rate over at least eight time intervals. Uses a leaky bucket to delay traffic, which achieves a smoothing effect.	Propagates bursts. Does no smoothing.
Advantages	Less likely to drop excess packets since excess packets are buffered. (Buffers packets up to the length of the queue. Drops may occur if excess traffic is sustained at high rates.) Typically avoids retransmissions due to dropped packets.	Controls the output rate through packet drops. Avoids delays due to queuing.
Disadvantages	Can introduce delay due to queuing, particularly deep queues.	Drops excess packets (when configured), throttling TCP window sizes and reducing the overall output rate of affected traffic streams. Overly aggressive burst sizes may lead to excess packet drops and throttle the overall output rate, particularly with TCP-based flows.
Optional Packet Remarking	No	Yes (with legacy CAR feature).

2.3 Traffic Policing

Traffic policing allows us to control the maximum rate of traffic sent or received on an interface, and to partition a network into multiple priority levels or class of service (CoS). The Traffic Policing feature manages the maximum rate of traffic through a token bucket algorithm. The token bucket algorithm can use the user-configured values to determine the maximum rate of traffic allowed on an interface at a given moment in time. The token bucket algorithm is affected by all traffic entering or leaving (depending on where the traffic policy with Traffic Policing configured) and is useful in managing network bandwidth in cases where several large packets are sent in the same traffic stream.

The token bucket algorithm provides users with three actions for each packet: a conform action, an exceed action, and an optional violate action. Traffic entering the interface with Traffic Policing configured is placed in to one of these categories. Within these three categories, users can decide packet treatments. For instance, packets that conform can be configured to be transmitted, packets that exceed can be configured to be sent with a decreased priority, and packets that violate can be configured to be dropped.

Traffic Policing is often configured on interfaces at the edge of a network to limit the rate of traffic entering or leaving the network. In the most common Traffic Policing configurations, traffic that conforms is transmitted and traffic that exceeds is sent with a decreased priority or is dropped. Users can change these configuration options to suit their network needs.

2.3.1 Benefits

The Benefits of Policing include:

- *Bandwidth Management Through Rate Limiting:* Traffic policing allows us to control the maximum rate of traffic sent or received on an interface. Traffic policing is often configured on interfaces at the edge of a network to limit traffic into or out of the network. Traffic that falls within the rate parameters is sent, whereas traffic that exceeds the parameters is dropped or sent with a different priority.
- *Packet Marking Through IP Precedence, QoS Group, and DSCP Value Setting:* Packet marking allows us to partition our network into multiple priority levels or classes of service (CoS), as follows:
 - Use traffic policing to set the IP precedence or differentiated services code point (DSCP) values for packets entering the network. Networking devices within our network can then use the adjusted IP Precedence values to determine how the traffic should be treated. For example, the DWRED feature uses the IP Precedence values to determine the probability that a packet will be dropped.

- Use traffic policing to assign packets to a QoS group. The router uses the QoS group to determine how to prioritize packets.

2.3.2 Restrictions

The following restrictions apply to the Traffic Policing feature:

- On a Cisco 7500 series router, traffic policing can monitor CEF switching paths only. In order to use the Traffic Policing feature, CEF must be configured on both the interface receiving the packet and the interface sending the packet.
- On a Cisco 7500 series router, traffic policing cannot be applied to packets that originated from or are destined to a router.
- Traffic policing can be configured on an interface or a subinterface.
- Traffic policing is not supported on the following interfaces:
 - Fast Ether Channel
 - Tunnel
 - PRI
 - Any interface on a Cisco 7500 series router that does not support CEF

2.4 Traffic Shaping

Traffic shaping allows us to control the traffic going out an interface in order to match its flow to the speed of the remote target interface and to ensure that the traffic conforms to policies contracted for it. Thus, traffic adhering to a particular profile can be shaped to meet downstream requirements, thereby eliminating bottlenecks in topologies with data-rate mismatches.

Cisco IOS QoS software has three types of traffic shaping: GTS, class-based, and FRTS. All three of these traffic shaping methods are similar in implementation, though their CLIs differ somewhat and they use different types of queues to contain and shape traffic that is deferred. In particular, the underlying code that determines whether enough credit is in the token bucket for a packet to be sent or whether that packet must be delayed is common to both features. If a packet is deferred, GTS and Class-Based Shaping use a weighted fair queue to hold the delayed traffic. FRTS uses either a custom queue or a priority queue for the same, depending on what we have configured.

2.4.1 Traffic Shaping and Rate of Transfer

Traffic shaping limits the rate of transmission of data. We can limit the data transfer to one of the following:

- A specific configured rate
- A derived rate based on the level of congestion

As mentioned, the rate of transfer depends on these three components that constitute the token bucket: burst size, mean rate, measurement (time) interval. The mean rate is equal to the burst size divided by the interval.

When traffic shaping is enabled, the bit rate of the interface will not exceed the mean rate over any integral multiple of the interval. In other words, during every interval, a maximum of burst size can be sent. Within the interval, however, the bit rate may be faster than the mean rate at any given time.

One additional variable applies to traffic shaping: Be size. The Excess Burst size corresponds to the number of non-committed bits—those outside the CIR—that are still accepted by the Frame Relay switch but marked as discard eligible (DE). In other words, the Be size allows more than the burst size to be sent during a time interval in certain situations. The switch will allow the packets belonging to the Excess Burst to go through but it will mark them by setting the DE bit. Whether the packets are sent depends on how the switch is configured.

When the Be size equals 0, the interface sends no more than the burst size every interval, achieving an average rate no higher than the mean rate. However, when the Be size is greater than 0, the interface can send as many as $B_c + B_e$ bits in a burst, if in a previous time period the maximum amount was not sent. Whenever less than the burst size is sent during an interval, the remaining number of bits, up to the Be size, can be used to send more than the burst size in a later interval.

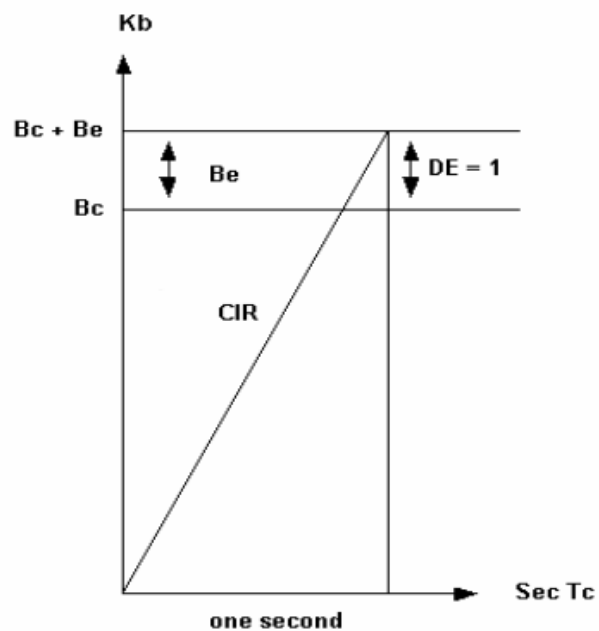


Figure 2.10

2.4.2 Traffic Shaping Parameters

Every implementation of traffic shaping first requires a series of calculation in order to determine the Shaping Parameters depending on the strategy of implementation. The various traffic shaping parameters that can be used are listed below. They will be discussed in detail in the following sections:

- CIR = committed information rate (= mean time)

- EIR = excess information rate = B_e
- TB = token bucket (= $B_c + B_e$)
- B_c = committed burst size (= sustained burst size)
- B_e = excess burst size
- DE = discard eligibility
- T_c = measurement interval
- AR = access rate corresponding to the rate of the physical interface (Ex. – for a T1, the AR is approximately 1.5 Mbps).

All these parameters are based on the token bucket mechanism discussed earlier in this chapter. The relationship between these parameters is shown by the graph in figure 2.10 (previous page).

2.4.3 Traffic Shaping and Queuing

*T*raffic shaping smoothes traffic by storing traffic above the configured rate in a queue.

When a packet arrives at the interface for transmission, the following sequence happens:

1. If the queue is empty, the arriving packet is processed by the traffic shaper.
 - a. If possible, the traffic shaper sends the packet.
 - b. Otherwise, the packet is placed in the queue.
2. If the queue is not empty, the packet is placed in the queue.
3. When packets are in the queue, the traffic shaper removes the number of packets it can send from the queue every time interval.

2.5 Chapter 3 - Preview

*I*n chapter 3, we will discuss the Lab setup used by us in testing and demonstrating the various traffic shaping techniques as a part of our major objective. The diagrammatic representations of the logical and physical topologies will give a distinct idea of the setup and thus will help in enhancing the comfort in understanding the implementations of the shaping techniques for the reader.

The following chapters will use this topology as a basis for all the experimental presentations. The device names and the interface addresses will remain constant throughout the report.

.....

CHAPTER 3

The Infrastructure

3.1 Scoop

This Chapter embodies the SECOND phase of the project's Life cycle and unarguably the most important of all other phases. The major challenge in this phase was to simulate a proficient real-time model which gives a commendable imitation of an end-to-end Client-ISP-Client scenario. Another concern was to make sure that the network can be customized and tailored with ease so as to create various scenarios that can simulate potential business requirements and thus, devise the most suitable Traffic Shaping solutions to implement the same.

After analyzing the requirements and the goals of the project and after careful examination of the available resources and the choices, the logical network shown in Figure 3.1 was contrived.





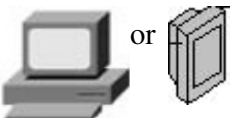


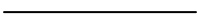

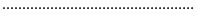
As is clearly depicted by the figure, this setup has a client called MINT Inc. who has offices in two different cities and the data and the bandwidth requirements are illustrated by the Voice and Video traffic that is sent along with the rest of the data traffic, between the two office locations. The key involvement of the client is to make effective usage of the available bandwidth. The client has purchased a frame-relay link from an ISP called TELUS. The Service Level Agreement with the ISP will offer optimized solution for making effective usage of the available bandwidth ensuring reliable voice quality and real-time video streaming between the two office locations over the frame relay cloud.

The following sections will elaborate the network topology, the devices and the protocols used. We will also discuss about the tools used by use to gauge the performances as well as the results.

3.2 The Blueprint

The topology discussed in this section was used as the basis for the all the experiments carried out in this project. Before we explain the actual topology lets quickly go through the icons used in this report (Table 3-1) to avoid any confusions.

Table 3-1 - The icons used in this report.

<i>S.No</i>	<i>Symbol</i>	<i>Description</i>
1		Router
2		Voice Enabled Router
3		Frame Relay Switch
4		Layer 2 Switch
5		Video server
6		Laptop / Test terminal
7		Cisco 7960 IP phone
8		Line: Ethernet
9		Line: Serial
10		Line: Frame Relay Virtual Connection

3.2.1 Logical Topology

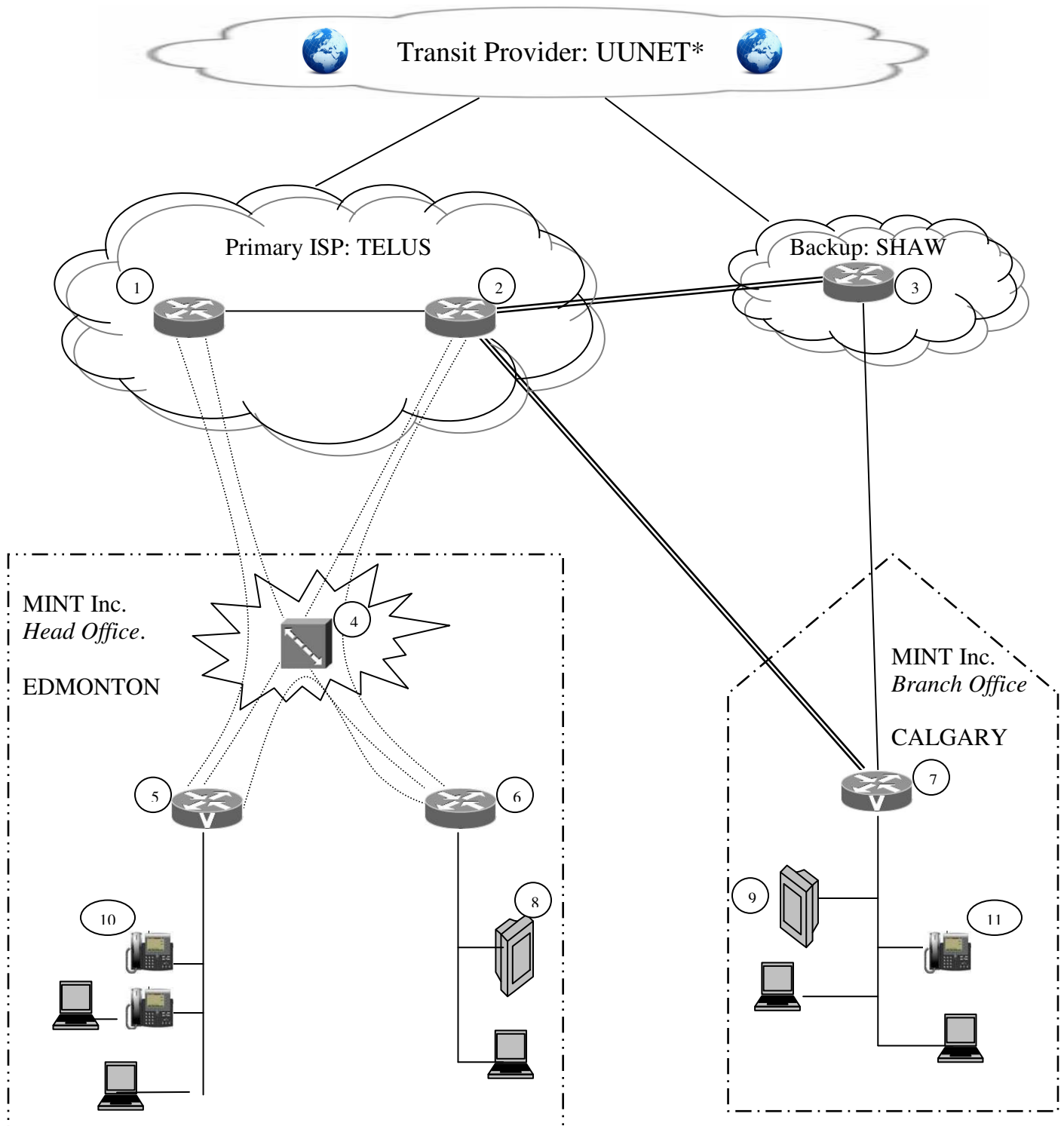


Figure 3.1

*Note: The transit provider, UUNET is not a part of the actual model. It is just shown to portray a possibility in the real-time environment.

3.2.2. The Physical Topology

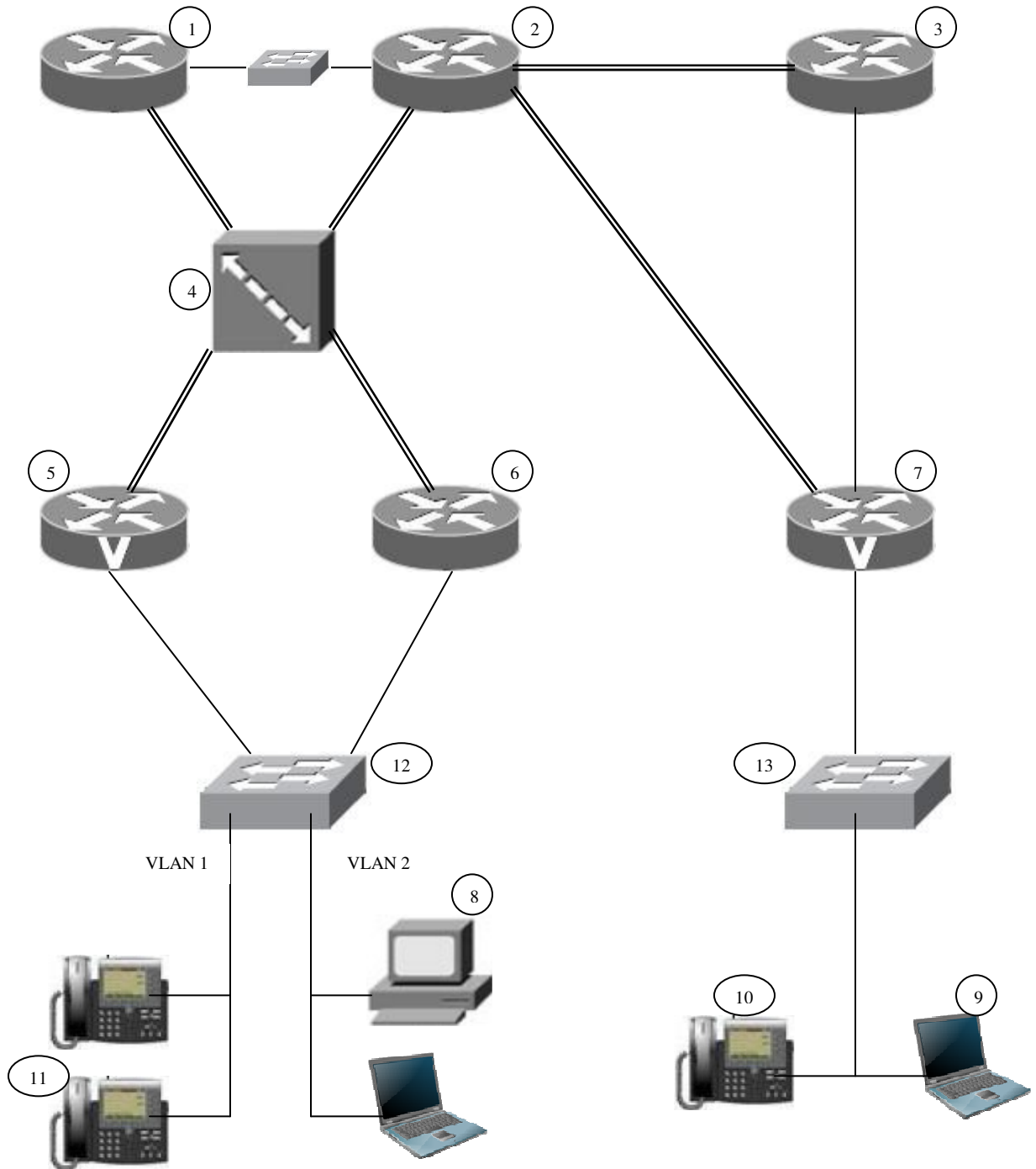


Figure 3.2

3.3 The Device Catalogue

The table below archives the brief description of each networking component that the model is comprised of. The Serial numbers used in this table correspond to the numbers in the circle in the previous figures. (Please refer back to Figures 3.1 and 3.2)

Table 3-2 – Functions of the network components used in the Model.

<i>S. No.</i>	<i>Device name</i>	<i>Description</i>
1	Telus2	<p>Telus2 is a Cisco 2800 series Integrated Services router (for small and medium business). In our scenario it plays the role of one of the Primary ISP (Telus) routers. This router is designated as a backup for Telus1 router and has two WAN links viz.</p> <ul style="list-style-type: none"> - A High speed Gb-Ethernet link to Telus1 [Port - GigabitEthernet 0/0] - A Serial link to the FR-SW that hosts two Frame Relay PVCs connecting to MINT-EDM1 and MINT-EDM2 [Port - Serial0/0/0.1 and Serial0/0/0.2]
2	Telus1	<p>Telus1 is a Cisco 3600 Series router; from the family of modular, multi-service access platforms for medium and large-sized offices and smaller Internet Service Providers. In our scenario, it plays the role of one of the Primary ISP (Telus) routers. This router hosts the primary connection for the Head-Office Location to the branch office location. It has four WAN links, viz.:</p> <ul style="list-style-type: none"> - An Ethernet link to Telus2 [Port – Ethernet 1/0] - A Serial link to the FR-SW that hosts two Frame Relay PVCs connecting to MINT-EDM2 and MINT-EDM1 [Port - Serial0/0.2 and Serial0/0.1] - A serial link to MINT-CAL router in the Branch Location at Calgary. [Port – Serial 0/1] - A serial link to the Backup ISP, Shaw’s router [Port - Serial 0/2]

3	Shaw	<p>Telus2 is again, a Cisco 2800 series Integrated Services router (for small and medium business). It is the ISP router for the Backup ISP named SHAW and plays the role of backing up the Primary ISP in case it goes down. It hosts the following two WAN links:</p> <ul style="list-style-type: none"> - A Serial link to Telus1 router at the primary ISP [Port – Serial s0/0/0] - A high speed Gb-Ethernet link to the MINT-CAL router in the Branch office location. [Port – GigabitEthernet 0/1]
4	FR-SW	<p>FR-SW is also a Cisco 3600 Series multi-service access platform router for medium and large-sized offices and smaller Internet Service Providers. It is used as a Frame Relay Switch that establishes a Frame Relay cloud between the Primary ISP and the Head Office location. It has the following four serial connections:</p> <ul style="list-style-type: none"> - A Serial link to the Telus1, that hosts two Frame Relay PVCs connecting to MINT-EDM2 and MINT-EDM1 [Port – Serial1/0] - A Serial link to the Telus2, that hosts two Frame Relay PVCs connecting to MINT-EDM1 and MINT-EDM2 [Port – Serial1/1] - A Serial link to the MINT-EDM1, that hosts three Frame Relay PVCs connecting to Telus1, Telus2 and MINT-EDM2. [Port – Serial1/2] - A Serial link to the MINT-EDM2, that hosts three Frame Relay PVCs connecting to Telus1, Telus2 and MINT-EDM1. [Port – Serial1/3]
5	MINT-EDM1	<p>MINT-EDM1 is a Cisco 2600 series Multi-service Platform router. It is a modular multi-service access router that provides flexible LAN and WAN configurations, multiple security options, and a range of high-performance processors. In our Lab model, it is used as a client side equipment, hosting Multi-service Voice and Data integration for the Head-office location. It connects Cisco 7960 IP phones across the WAN cloud by implementing Frame-relay based packet telephony solutions for the voice traffic between the two client equipments viz. MINT-EDM1 and MINT-CAL. There are Two live physical</p>

		<p>connections on this device:</p> <ul style="list-style-type: none"> - A Serial link to the FR-SW that hosts three Frame Relay PVCs connecting to Telus1, Telus2 and MINT-EDM2 [Port – Serial0/1.2, Serial0/1.1 and Serial0/1.3] - An Ethernet link to the Cisco 3500 Layer2 switch, connecting to the local subnet. [Port – Ethernet 0/1]
6	MINT-EDM2	<p>MINT-EDM2 is also a Cisco 2600 series Multi-service Platform router. It is used as client-side equipment, hosting Multi-service Video and Data integration for the Head-Office location at Edmonton. It hosts a Video server (Windows VLC Media Server) and transmits the video stream across the network to the branch location (MINT-CAL). The physical connections are:</p> <ul style="list-style-type: none"> - A Serial link to the FR-SW that hosts three Frame Relay PVCs connecting to Telus1, Telus2 and MINT-EDM1 [Port – Serial0/1.1, Serial0/1.2 and Serial0/1.3] - An Ethernet link to the Cisco 3500 Layer2 switch, connecting to the local subnet. [Port – Ethernet 0/1]
7	MINT-CAL	<p>MINT-CAL is again a Cisco 2600 series Multi-service Platform router. It is used as a client side equipment, hosting Multi-service Voice, Video and Data integration for the Branch-Office location at Calgary. It ensures delivery of concurrent data, voice and video services. It hosts Cisco IP phones as well as a Video client (Windows VLC Media client). The physical connections are:</p> <ul style="list-style-type: none"> - A Serial connection to the primary ISP via Telus1 router. This is the primary link. [Port – Serial 0/1] - A high speed Ethernet connection to the Backup ISP via Shaw router. This is a fallback link. [Port Ethernet 0/1] - An Ethernet connection to a Cisco 3500 switch connecting to the local subnet [Port – Ethernet 0/0]
8	Video Server	<p>It is a Windows Server 2003 machine which is running the Windows VLC Media Server, used to stream video data across the network to the VLC Media Client running at the Branch Office location. This setup is used to simulate the</p>

		Video streaming activity in the company's network.
9	Video Client	This is any Windows Workstation or Notebook running the Windows VLC Media Client and accepting the video stream from the Video Server at the Head Office location. This setup is used to simulate the Video streaming activity in the company's network.
10, 11	7960 IP phones	These are the full featured Cisco IP phones that provide full voice communication over the same data network that is used by the other applications. We have used these phones at both the locations with dual lines on each phone i.e. each phone has two distinct phone numbers assigned to it. They are connected to the voice enabled routers MINT-EDM1 and MINT-CAL at their respective locations. The route through Telus1 is used as the primary link between the two voice peers.
12, 13	Switches	These switches are the Catalyst 3500 Series (#12) and Catalyst 3700 Series (#13) switches and are used in each location's local network to connect the various devices to their respective local gateways. The switch #12 at the Head office is split into two VLANs to separate the two internal subnets – one with the voice data and the other one with the video data. The switch #13 is basically a Layer 3 switch, however it is used only at Layer 2. It connects the video clients, IP phones and the other hosts within the same subnet to the MINT-CAL router.

3.4 *The Big Picture*

The Network Model used in the report is a manifestation of a real-time scenario in which an enterprise named MINT Inc. is an Academic Organization and has two office locations established in its domain – Head Office at Edmonton, and Branch Office at Calgary. The business needs of the organization demand enough bandwidth for its Voice and Video data integrated with raw data to be transmitted to / received from, the other location. MINT inc. has signed a Service Level Agreement with TELUS that guarantees reliable communication and a consistent bandwidth for all kinds of data by implementing sophisticated Traffic

Shaping solutions for Voice and Video. Now, let us talk about the strategies devised by TELUS to realize these commitments.

Telus is a well-known ISP in the country. It has established business ties with another ISP named SHAW that provides backup for the Transit connection to UUNET (The Transit Provider). TELUS has provided a FRAME RELAY link to the Head Office location of MINT Inc. Understanding the fact that MINT Inc.'s network is essentially branched into two subnets viz. Voice (MINT-EDM1) and Video (MINT-EDM2), it has provided two Frame Relay PVC connections to the Head Office LAN, one for each subnet. That is, one of these PVCs will carry Voice+Data traffic from MINT-EDM1 and the other will carry the Video+Data traffic from the MINT-EDM2. These traffic streams will pass through the Frame relay cloud followed by the Telus1 router in the ISP cloud, that will pass it to the Branch location for Intranet destination OR to the Transit Cloud for the Internet destination. Thus Telus1 is the primary path for all types of traffic.

Telus also implements an effective backup mechanism by providing a set of backup PVC connections to the Head office location from the Telus2 router. Thus, in case the serial link on Telus1 goes down, the data stream can failover to the backup link from Telus2. This mechanism is implemented by manipulating the BGP attributes. Detailed discussion about this is covered in section 3.6.1 of this report.

The Frame relay cloud has a bottleneck of 128Kbps at the FR-SW frame relay switch. This means that the effective data-rate, collaborative of all types of data traffic should be limited to 128Kbps. Thus the available bandwidth should be segregated, so as to allocate sufficient amount of channel to each PVC that will ensure reliable quality of service. In chapter 6 of this report, we will implement Frame Relay Traffic shaping on this part of the network. This implementation will allocate specific amount of bandwidth to the voice and the video channel and will thus ensure that each of these applications will get their fair share of the network resources.

Coming over to the Branch office location; it has a primary link to the 'Telus1' router at the TELUS ISP and a backup link to the 'Shaw' router at the backup ISP, SHAW. Both these links are capable enough to carry all kinds of traffic concurrently. We will still need to allocate a fair share to each of these. This side hosts a video client that accepts a video stream from the Video Server setup at the MINT-EDM2 router at the Head office. It also has an IP phone and other workstations communicating with the head office at the same time.

The Shaw ISP is a Backup ISP that is home to the 'Shaw' router. We will implement generic traffic shaping on this box to smooth the traffic flow across its peers. The major functionalities involved with it are:

- Backing up MINT-CAL, if its serial link to Telus1 is down
- Backing up TELUS if its link to the Transit goes down.

This should have given a fairly qualified idea of the whole picture. The following sections will provide additional information to explain the implementation. Next, we will have a quick look on the IP addressing scheme embraced in this LAB followed by the protocols and tools used to implement this scenario.

3.5 IP Addressing

Selecting a pertinent addressing scheme is very influential element of the process of the network design. A wisely selected address lot turn tables for a lot of issues that could have existed otherwise. In our network:

- TELUS has provided the address range of 100.0.0.0/8 to the company's network
- This address range is subnetted as follows:
 - o 100.4.0.0/16 is allocated for the Voice subnet at the Head-Office.
 - o 100.6.0.0/16 is allocated for the Video subnet at the Head-Office.
 - o 100.6.0.0/16 is allocated for the subnet at the Branch-Office.
 - o The subnets 100.3.0.0/30, 100.5.0.0/30, 100.1.10.0/30, 100.1.20.0/30, 100.2.10.0/30, 100.2.20.0/30 and 150.0.0.0/30 are used for the WAN links
- SHAW has provide an IP range of 200.0.0.0/16, but is not subnetted due to lack of usage.

The following table summarizes the IP addresses of almost all the network interfaces of all the devices being used in this Lab. We have also included the IP addresses received from the DHCP lease in valid cases.

Table 3-3 – IP Addressing Scheme

<i>S. No.</i>	<i>Hostname of Switch / Router / Workstation / IP Phone</i>	<i>Interface / Subinterface name</i>	<i>IP address with CIDR</i>	<i>DHCP Pool Exclusion</i>	<i>DHCP Pool Network</i>	<i>Default Gateway</i>
1	Telus1	Ethernet 1/0	100.0.0.1/30	NA	NA	Transit
		Serial 0/2	150.0.0.1/30	NA	NA	Transit
		Serial 0/1	100.5.0.1/30	NA	NA	Transit
		Serial0/0.1	100.1.20.1/30	NA	NA	Transit
		Serial0/0.2	100.1.10.1/30	NA	NA	Transit
2	Telus2	GbEthernet0/0	100.0.0.2/30	NA	NA	Transit
		Serial0/0/0.1	100.2.10.1/30	NA	NA	Transit
		Serial0/0/0.2	100.2.20.1/30	NA	NA	Transit

3	MINT-EDM1	Serial 0/1.1	100.2.10.2/30	NA	NA	100.2.10.1
		Serial 0/1.2	100.1.10.2/30	NA	NA	100.1.10.1
		Serial 0/1.3	100.3.0.2/30	NA	NA	100.1.10.1
		Ethernet 0/1	100.4.0.1/16	100.4.0.1 to 100.4.0.5	100.4.0.0 /16	100.1.10.1
4	MINT-EDM2	Serial 0/1.1	100.1.20.2/30	NA	NA	100.1.20.1
		Serial 0/1.2	100.2.20.2/30	NA	NA	100.2.10.1
		Serial 0/1.3	100.3.0.1/30	NA	NA	100.1.20.1
		Ethernet 0/1	100.6.0.1/16	NA	NA	100.1.20.1
5	Shaw	Serial 0/0/0	150.0.0.2/30	NA	NA	Transit
		GbEthernet0/1	200.0.0.1/30	NA	NA	Transit
6	MINT-CAL	Serial 0/1	100.5.0.2/30	NA	NA	100.5.0.1
		Ethernet 0/1	200.0.0.2/30	NA	NA	200.0.0.1
		Ethernet 0/0	100.8.0.1/16	100.8.0.1 to 100.8.0.5	100.8.0.0 /16	100.5.0.1
7	VLC Media Server	Ethernet	100.6.0.3	NA	NA	100.6.0.1
8	VLC Media Client	Ethernet	100.8.0.8	100.8.0.1 to 100.8.0.5	100.8.0.0 /16	100.8.0.1
9	IP Phone 1 at the Head Office	Ethernet	100.4.0.6	100.4.0.1 to 100.4.0.5	100.4.0.0 /16	100.4.0.1

10	IP Phone 1 at the Head Office	Ethernet	100.4.0.7	100.4.0.1 to 100.4.0.5	100.4.0.0 /16	100.4.0.1
11	IP Phone at the Branch Office	Ethernet	100.8.0.6	100.8.0.1 to 100.8.0.5	100.8.0.0 /16	100.8.0.1

3.6 Protocols Used

In this section we will talk about the various protocols used in the scenario, the objective that they are supposed to bring off and how they are implemented. These protocols can be categorized as Layer 1 signaling protocols, Layer 2 data link encapsulation protocols, Layer 3 routing protocols or Layer 4 Transport control protocols. (Protocols at higher layers are not covered in details). Lets discuss them one by one.

3.6.1 Ethernet

Ethernet defines a number of wiring and signaling standards for the physical layer (Layer1), through means of network access at the Media Access Control (MAC)/Data Link Layer (Layer 2), and a common addressing format. It has been standardized as IEEE 802.3. The combination of the twisted pair versions of Ethernet for connecting end systems to the network, along with the fiber optic versions for site backbones, has become the most widespread wired LAN technology. A scheme known as carrier sense multiple access with collision detection (CSMA/CD) is used to govern the way the peers shared the channel. This protocol is employed in all the LAN segments in our Network Model and in some WAN connections as well.

3.6.2 Frame Relay

Frame Relay is a high-performance WAN protocol that operates at the physical and data link layers of the OSI reference model. It consists of an efficient data transmission technique

used to send digital information quickly and cheaply in a relay of frames to one or many destinations from one or many end-points. Frame relay is commonly implemented for voice and data as an encapsulation technique, used between local area networks over a wide area network. Each end-user gets a private line (or leased line) to a frame-relay node. The frame-relay network handles the transmission over a frequently-changing path transparent to all end-users.

Thus, frame relay is aimed at a telecommunication service for cost-efficient data transmission for intermittent traffic between local area networks and between end-points in a wide area network. It thus proved to be an excellent selection to implement in our network model.

To summarize it yet again, our Network Provider has enforced a Frame Relay cloud to allocate two leased lines to each router in the Head office location. One of these routers will be transmitting Voice+Data (MINT-EDM1) and the other will be sending Data+Video streams (MINT-EDM2) to the branch office location. Each router will use only one of the two lines allocated to them and the other one will be used as a backup. Additionally, both the primary leased lines are homed at the Telus1 router and the backup lines originate from the Telus2 router in the ISP cloud. The ISP cloud is comprised of the FR-SW router, which is a Multiservice Cisco 3600 series router. The configuration files for this device are included in APPENDIX – I of this report.

3.6.3 Border Gateway Protocol

Border Gateway Protocol (BGP) is an inter-domain routing protocol designed to provide loop-free routing between separate routing domains that contain independent routing policies (autonomous systems). It is designed to run over a reliable transport protocol; it uses TCP (Port 179) as the transport protocol because TCP is a connection-oriented protocol. The destination TCP port is assigned 179, and the local port assigned a random port number. BGP is mainly used to connect a local network to an external network to gain access to the Internet or to connect to other organizations. When connecting to an external organization, external BGP (eBGP) peering sessions are created, whereas BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

BGP uses a path-vector routing algorithm to exchange network reachability information with other BGP speaking networking devices. Network reachability information is exchanged between BGP peers in routing updates. It contains the network number, path specific attributes, and the list of autonomous system numbers that a route must transit through to reach a destination network.

But before this negotiation starts, the BGP peers should establish a neighbor relationship using a TCP connection. BGP neighbor negotiation proceeds through different stages before the connection is fully established. Figure 3.3 illustrates a simplified finite state machine that highlights the major events in the process with an indication of messages (OPEN, KEEPALIVE, NOTIFICATION) sent to the peer in the transition from one state to the other.

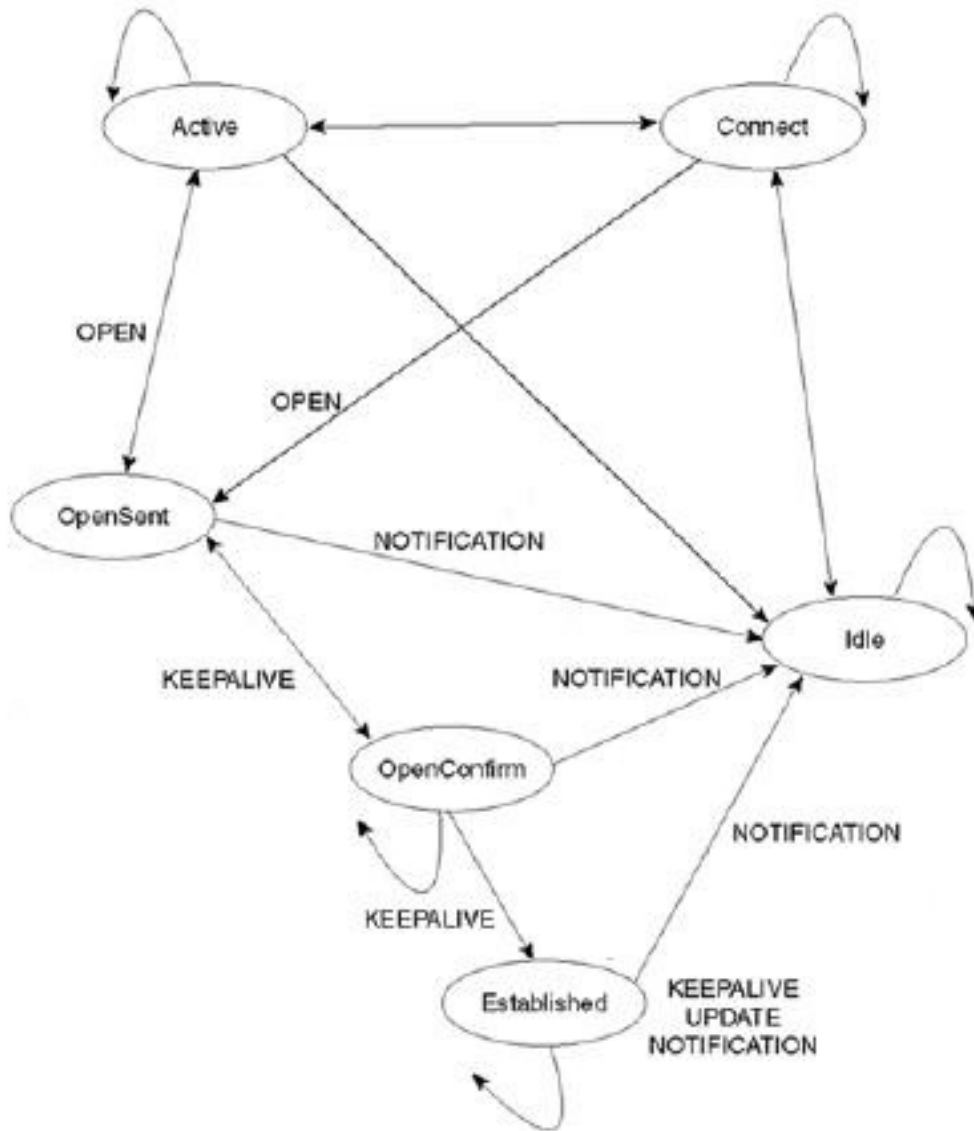


Figure 3.3

BGP starts exchanging UPDATE packets with its peers after entering the “Established” state, which means that the neighbor relationship has been established between the BGP speakers. They can now start exchanging full routing information over the TCP connection. Note that when changes to the routing table are detected, the BGP routers send to their neighbors only those routes that have changed.

[A] BGP Attributes

Routes learned via BGP have associated properties that are used to determine the best route to a destination when multiple paths exist to a particular destination. These properties are referred to as BGP attributes, and an understanding of how BGP attributes influence route selection is required for the design of robust networks. Lets talk in brief about these attributes explaining how they have been implemented in our Lab model as well:

a) Weight Attribute

Weight is a Cisco-defined attribute that is local to a router. The weight attribute is not advertised to neighboring routers. According to this attribute, if the router learns about more than one route to the same destination, the route with the highest weight will be preferred.

b) Local Preference Attribute

Unlike the weight attribute, the local preference attribute is propagated throughout the local AS. If there are multiple exit points from the local AS, the local preference attribute is used to select the preferred exit point for a specific route. The exit point with HIGHER value of Local Preference is the preferred path. In our network model we have used this attribute to determine the preferred exit point from the Autonomous System at the head office location (AS = 1001).

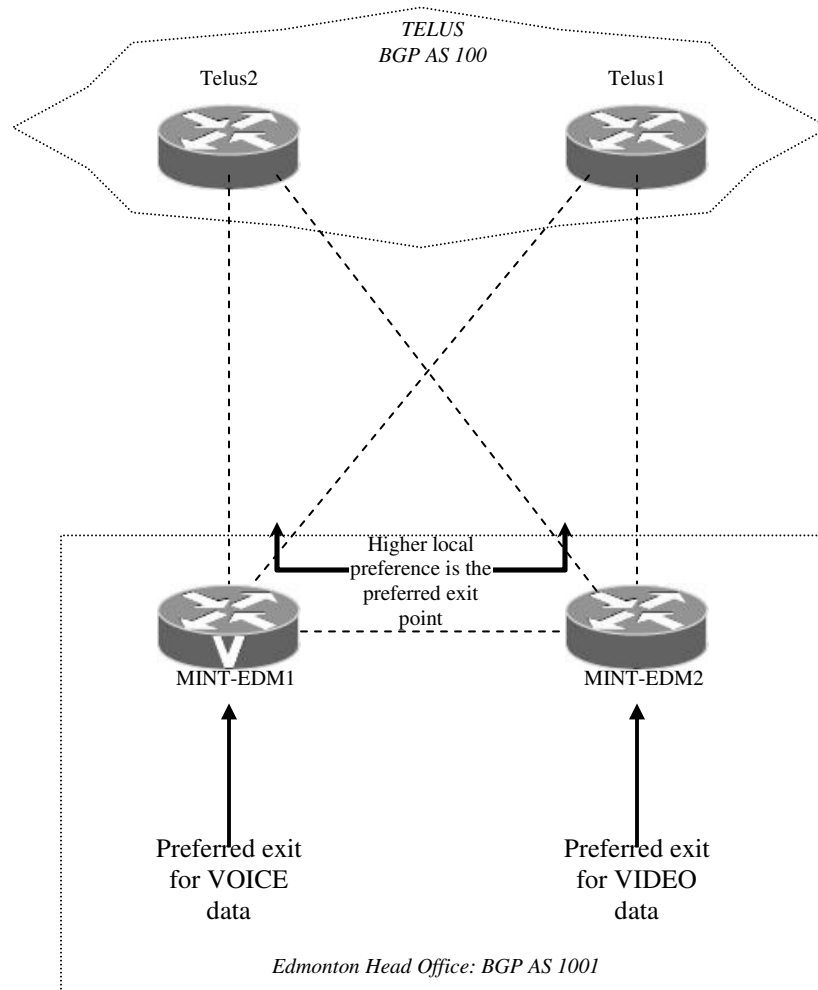


Figure 3.4

As is apparent from the Figure below, Head office location has two exit points viz. – MINT-EDM1 and MINT-EDM2 routers. Now our condition is that for the voice data, MINT-EDM1 should be the preferred exit and for the video data, MINT-EDM2 should be the preferred exit point. We thus configure both the routers to prefer itself to be the exit-point for the data

originated by it. Here is a sample configuration done on the MINT-EDM1 router to prefer itself to be the exit for the voice data originated by it:

```
!  
//Output reduced  
!  
neighbor 100.1.10.1 remote-as 100  
  neighbor 100.1.10.1 route-map lpref4primary out  
  neighbor 100.2.10.1 remote-as 100  
  neighbor 100.2.10.1 route-map lpref4backup out  
  neighbor 100.3.0.1 remote-as 1001  
  no auto-summary  
!  
ip http server  
ip classless  
!  
!  
!  
route-map lpref4primary permit 10  
  set local-preference 200  
!  
route-map lpref4backup permit 10  
  set local-preference 90  
!  
//Output reduced  
!
```

Similarly, the following configuration was done on the MINT-EDM2 router to prefer itself to be the exit for the Video data originated by it

```
!  
//Output reduced  
!  
neighbor 100.1.20.1 remote-as 100  
  neighbor 100.1.20.1 route-map lpref4primary out  
  neighbor 100.2.20.1 remote-as 100  
  neighbor 100.2.20.1 route-map lpref4backup out  
  neighbor 100.3.0.2 remote-as 1001  
  no auto-summary  
!  
ip http server  
ip classless  
!  
!  
route-map lpref4primary permit 10  
  set local-preference 200  
!  
route-map lpref4backup permit 10  
  set local-preference 90  
!  
//Output reduced  
!
```


Basically, we are just implementing a *route-map* to manipulate the local preference of each box. So, when data is received at the MINT-EDM1 router, the decision has to be made to decide the exit point. It then alters its local preference from 100 (default) to 200 and changes MINT-EDM2's local preference to 90. The reverse happens in case of MINT-EDM2.

(c) Multi-Exit Discriminator Attribute

The multi-exit discriminator (MED) or metric attribute is used as a suggestion to an external AS regarding the preferred route into the AS that is advertising the metric. The term suggestion is used because the external AS that is receiving the MEDs may be using other BGP attributes for route selection. MED basically represents the metric of a link and thus lower MED link is the preferred route. As shown in figure 3.5, in our LAB model the ISP routers use the MED attribute to announce to the Head-Office location (AS=1001) that Telus1 is the preferred router for all the data flowing into the ISP's autonomous system (AS = 100)

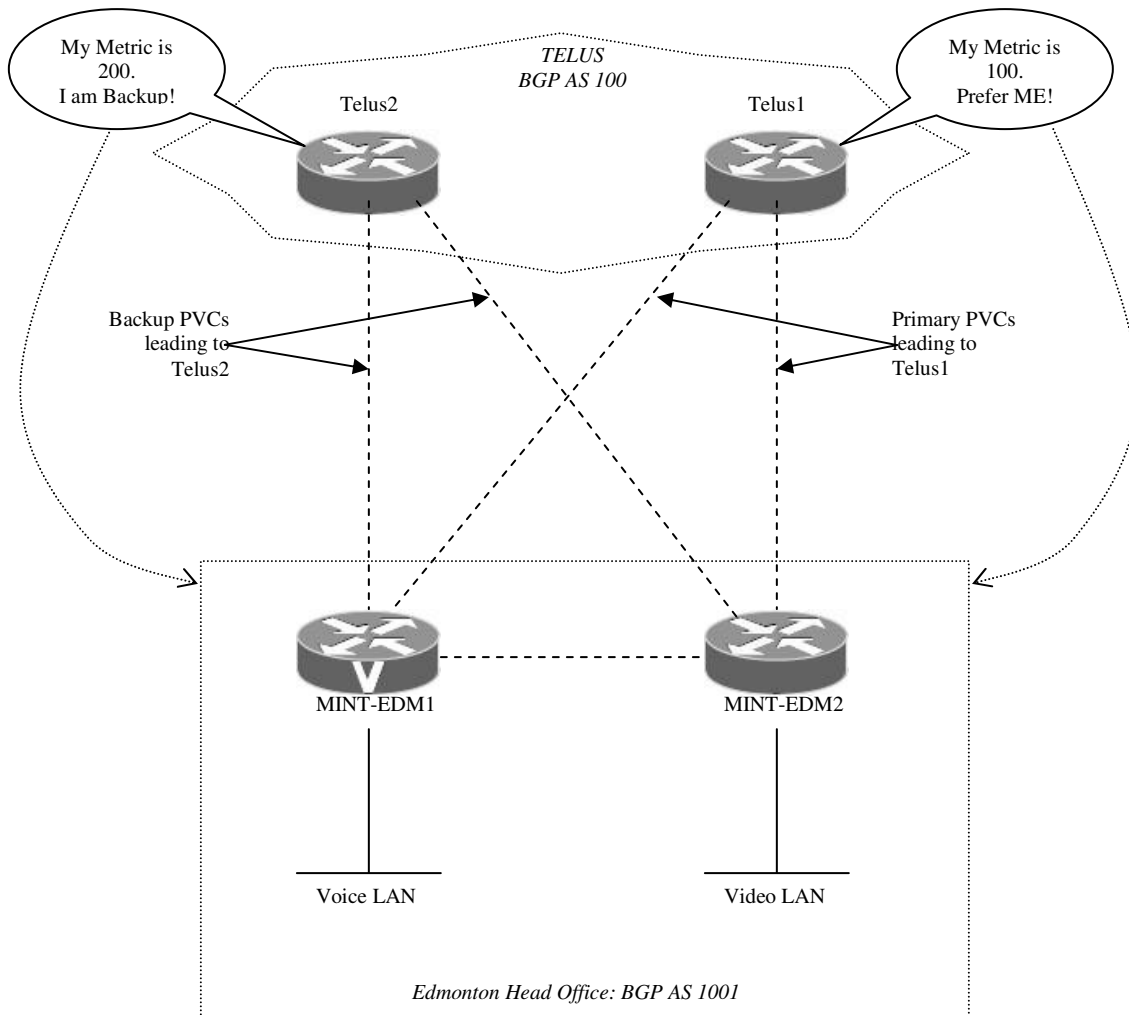


Figure 3.5

The MED is manipulated in both Telus1 and Telus2 routers using the following commands:

Telus1

```
!  
//output reduced  
!  
neighbor 100.1.10.2 description Head-Office Voice Gateway (Primary  
Link)  
  neighbor 100.1.10.2 default-originate  
  neighbor 100.1.10.2 route-map med4primary out  
  neighbor 100.1.20.2 remote-as 1001  
  neighbor 100.1.20.2 description Head-Office Server Gateway (Primary  
Link)  
  neighbor 100.1.20.2 default-originate  
!  
ip classless  
ip http server  
!  
route-map med4primary permit 10  
  set metric 100  
!  
//output reduced  
!
```

Telus2

```
!  
//output reduced  
!  
  neighbor 100.2.10.2 description Head-Office Voice Gateway  
  neighbor 100.2.10.2 default-originate  
  neighbor 100.2.10.2 route-map med4backup out  
  neighbor 100.2.20.2 remote-as 1001  
  neighbor 100.2.20.2 description Head-Office Video Server  
  neighbor 100.2.20.2 default-originate  
  neighbor 100.2.20.2 route-map med4backup out  
  no auto-summary  
!  
ip classless  
!  
!  
ip http server  
no ip http secure-server  
!  
!  
route-map med4backup permit 10  
  set metric 200  
!  
//output reduced  
!
```

Note: The “*default originate*” command is used to advertise a default route from the ISP routers to the client routers. This means that the ISP routers tell the client routers that any route that is not resolved by the client routers will be sent to the ISP router(s).

(d) Origin

The origin attribute indicates how BGP learned about a particular route. It tells us if the route is internal to the originating AS. The origin attribute can have one of three possible values:

- IGP (i) —The route is interior to the originating AS. This value is set when the network router configuration command is used to inject the route into BGP.
- EGP (e) —The route is learned via the Exterior Border Gateway Protocol (EBGP).
- Incomplete (?) —The origin of the route is unknown or learned in some other way. An origin of incomplete occurs when a route is redistributed into BGP.

(d) AS_path Attribute

When a route advertisement passes through an autonomous system, the AS number is added to an ordered list of AS numbers that the route advertisement has traversed. Figure 39-5 shows the situation in which a route is passing through three autonomous systems.

AS1 advertises a route to AS2 and AS3, with the AS_path attribute equal to {1}. AS3 will advertise back to AS1 with AS-path attribute {3,1}, and AS2 will advertise back to AS1 with AS-path attribute {2,1}. AS1 will reject these routes when its own AS number is detected in the route advertisement. This is the mechanism that BGP uses to detect routing loops. AS 2 and AS 3 propagate the route to each other with their AS numbers added to the AS_path attribute. These routes will not be installed in the IP routing table because AS 2 and AS 3 are learning a route from AS 1 with a shorter AS_path list.

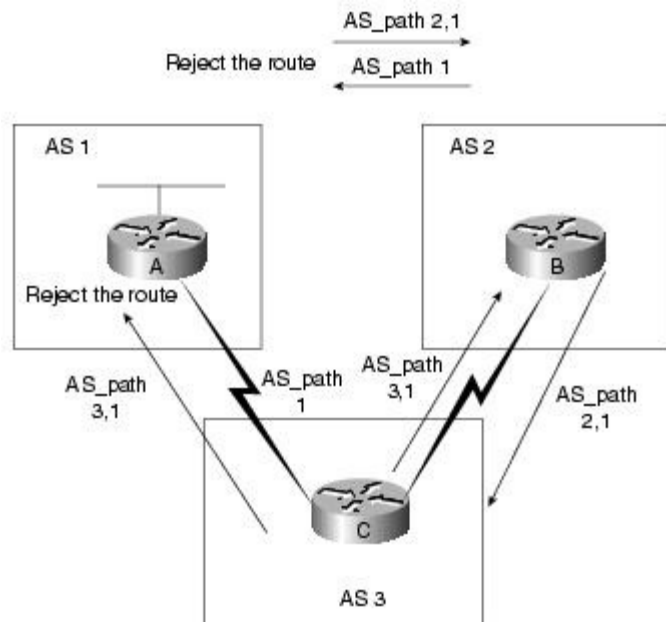


Figure 3.6

(e) Next-Hop Attribute

The EBGP next-hop attribute is the IP address that is used to reach the advertising router. For EBGP peers, the next-hop address is the IP address of the connection between the peers. For IBGP, the EBGP next-hop address is carried into the local AS.

(f) Community Attribute

The community attribute provides a way of grouping destinations, called communities, to which routing decisions (such as acceptance, preference, and redistribution) can be applied.

Route maps are used to set the community attribute. Predefined community attributes are listed here:

- no-export—Do not advertise this route to EBGp peers.
- no-advertise—Do not advertise this route to any peer.
- internet—Advertise this route to the Internet community; all routers in the network belong to it.

We have not used this attribute in our network model.

[B] BGP Path Selection

BGP could possibly receive multiple advertisements for the same route from multiple sources. BGP selects only one path as the best path. When the path is selected, BGP puts the selected path in the IP routing table and propagates the path to its neighbors. BGP uses the following criteria, in the order presented, to select a path for a destination:

- If the path specifies a next hop that is inaccessible, drop the update.
- Prefer the path with the largest weight.
- If the weights are the same, prefer the path with the largest local preference.
- If the local preferences are the same, prefer the path that was originated by BGP running on this router.
- If no route was originated, prefer the route that has the shortest AS_path.
- If all paths have the same AS_path length, prefer the path with the lowest origin type (where IGP is lower than EGP, and EGP is lower than incomplete).
- If the origin codes are the same, prefer the path with the lowest MED attribute.
- If the paths have the same MED, prefer the external path over the internal path.
- If the paths are still the same, prefer the path through the closest IGP neighbor.
- Prefer the path with the lowest IP address, as specified by the BGP router ID.

[C] Summarization

Summarization is done to reduce the size of the routing table and helps the router to decrease the length of the search needed to find a match for the next hop, thus improving efficiency and network speeds. In our LAB, we have the TELUS ISP to send a summarized route to the SHAW ISP, thus reducing the size of the Shaw's routing table:

```
network 150.0.0.0 mask 255.255.255.252
aggregate-address 100.0.0.0 255.0.0.0 summary-only
```

3.6.4 Open Shortest Path First (OSPF)

The Open Shortest Path First (OSPF) protocol is a hierarchical interior gateway protocol (IGP) for routing in Internet Protocol, using a link-state in the individual areas that make up the hierarchy. A computation based on Dijkstra's algorithm is used to calculate the shortest

path tree inside each area. The algorithm places each router at the root of a tree and calculates the shortest path to each destination based on the cumulative cost required to reach that destination. Each router will have its own view of the topology even though all the routers will build a shortest path tree using the same link-state database.

Following is the list of some key features offered by OSPF that makes it the most abundantly used IGP:

- With OSPF, there is no limitation on the hop count.
- The intelligent use of VLSM is very useful in IP address allocation.
- OSPF uses IP multicast to send link-state updates. This ensures less processing on routers that are not listening to OSPF packets. Also, updates are only sent in case routing changes occur instead of periodically. This ensures a better use of bandwidth.
- OSPF has better convergence than other protocols like RIP. This is because routing changes are propagated instantaneously and not periodically.
- OSPF allows for better load balancing.
- OSPF allows for a logical definition of networks where routers can be divided into areas. This will limit the explosion of link state updates over the whole network. This also provides a mechanism for aggregating routes and cutting down on the unnecessary propagation of subnet information. It is clearly represented in Figure 3.8
- OSPF allows for routing authentication by using different methods of password authentication.
- OSPF allows for the transfer and tagging of external routes injected into an Autonomous System. This keeps track of external routes injected by exterior protocols such as BGP.

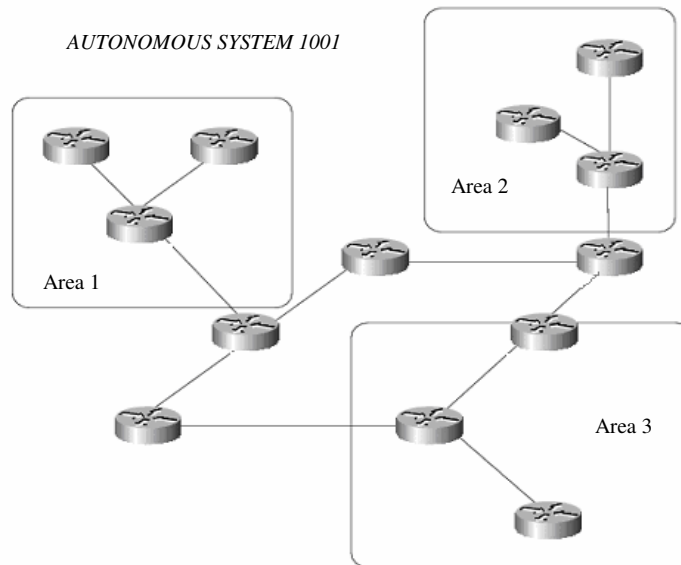


Figure 3.7

In our Lab we have four autonomous systems in total viz. AS100 (TELUS), AS1001 (Head-Office, Edmonton), AS1002 (Branch-Office, Calgary) and AS200 (SHAW). In each of these autonomous systems, we are running OSPF as the IGP. It is a single area implementation of OSPF. The following table lists the key commands used to configure OSPF

	Command	Purpose
Step 1	Router(config)# <code>router ospf process-id</code>	Enables OSPF routing, which places you in router configuration mode.
Step 2	Router(config-router)# <code>network ip-address wildcard-mask area area-id</code>	Defines an interface on which OSPF runs and define the area ID for that interface.

3.6.5 TCP and UDP

*T*CP (Transmission Control Protocol) is the most commonly used protocol on the Internet. The reason for this is because TCP offers error correction. When the TCP protocol is used there is a "guaranteed delivery." This is due largely in part to a method called "flow control." Flow control determines when data needs to be re-sent, and stops the flow of data until previous packets are successfully transferred. This works because if a packet of data is sent, a collision may occur. When this happens, the client re-requests the packet from the server until the whole packet is complete and is identical to its original.

UDP (User Datagram Protocol) is another commonly used protocol on the Internet. However, UDP is never used to send important data such as webpages, database information, etc; UDP is commonly used for streaming audio and video. Streaming media such as Windows Media audio files (.WMA), Real Player (.RM), and others use UDP because it offers speed! The reason UDP is faster than TCP is because there is no form of flow control or error correction. The data sent over the Internet is affected by collisions, and errors will be present. Remember that UDP is **only** concerned with speed. This is the main reason why streaming media is not high quality.

3.6.6 Real Time Protocol (RTP)

*T*he Real-time Transport Protocol (or RTP) defines a standardized packet format for delivering audio and video over the Internet. In our network, the voice and video data is considered to be Real Time data. Separate sessions are used for both audio and video. The main advantage of this separation is to make it possible to receive only one part of the transmission, commonly audio data, which lowers the total bandwidth. RTP does not have a

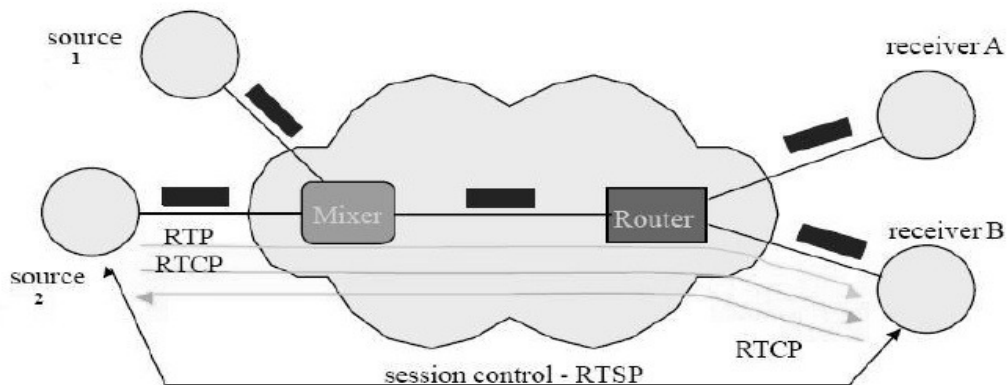


Figure 3.8

standard TCP or UDP port on which it communicates. The only standard that it obeys is that UDP communications are done via an even port and the next higher odd port is used for RTP Control Protocol (RTCP) communications. Although there are no standards assigned, RTP is generally configured to use ports 16384-32767. RTP can carry any data with real-time

characteristics, such as interactive audio and video. RTP uses UDP. It uses a special set of messages (RTCP) to exchange periodic reports in one RTP session, there is one media flow. Mixer is an intermediate system that combines RTP streams from different sources into a single stream. It can change the data format of the RTP packets.

3.6.7 Network Time Protocol (NTP)

*N*etwork Time Protocol (NTP) allows us to synchronize Cisco CME router to a single clock on the network, which is known as the clock master. NTP is disabled on all interfaces by default, but it is essential for Cisco CME.

The main characteristics of NTP are the following.

- Fully automatic, keeps continuously the synchronization.
- Suitable to synchronize one computer as well as a whole computer network.
- Available on almost every type of computer.
- Fault tolerant and dynamically auto-configuring.
- Carrying UTC time, independent of time zones and day-light saving time.
- Synchronization accuracy can reach 1 millisecond.

3.6.8 Skinny Client Protocol (SCCP)

*S*kinny Client Control Protocol (SCCP) is a Cisco proprietary protocol used between Cisco Call Manager and Cisco VOIP phones. It is also supported by some other vendors. It defines a simple and easy to use architecture, while the H.323 recommendations are quite an expensive system. An H.323 proxy can be used to communicate with the Skinny Client using the SCCP. In such a case the telephone is a skinny client over IP, in the context of H.323. The skinny client (i.e. an Ethernet Phone) uses TCP/IP to transmit and receive calls and RTP/UDP/IP to/from a Skinny Client or H.323 terminal for audio

With the SCCP architecture, the vast majority of the H.323 processing power resides in an H.323 proxy known as the Cisco Call Manager. The end stations (telephones) run what is called the Skinny Client, which consumes less processing overhead. The Client communicates with the Call Manager using connection-oriented (TCP/IP-based) communication to establish a call with another H.323-compliant end station. Once the Call Manager has established the call, the two H.323 end stations use connectionless (UDP/IP-based) communication for audio transmissions. Costs and overhead are thus reduced by confining the complexities of H.323 call setup to the Call Manager, and using the Skinny protocol for the actual audio communication into and out of the end stations.

3.7 The Toolkit

This section covers the distinct tools that we have used in this project. We will give the a brief overview of each with a little detailed discussion about the Cisco Call Manager Express.

3.7.1 Monitoring Tools

[A] Ethereal ver0.10.14

Ethereal is a protocol analyzer, or "packet sniffer" application, used for network troubleshooting, analysis, software and protocol development, and education. It has all of the standard features of a protocol analyzer. Most of the Traffic statistics were captured and tested using this tool.

[B] VLC Player ver3.4.5

VLC Player is a highly portable multimedia player, encoder, and streamer supporting many audio and video codecs and file formats as well as DVDs, VCDs, and various streaming protocols. It is able to stream over networks and to transcode multimedia files and save them into various different formats. We have used this as the Video server at one end and a video client at the other end. It also helped us to *Transcode* the video clips into lower resolution videos, to comply with the bandwidth restrictions at the frame relay cloud.

[C] Router IOS Debug commands

A series of "*debug*" and "*show*" commands have been used in our experiments to capture the results of our implementation. The outputs of these commands display a fairly organized, consolidated view of the configuration file that proves to be a useful source for troubleshooting as well.

3.7.2 Cisco CallManager Express

The Cisco CallManager Express is a solution embedded in Cisco IOS Software that provides call processing for Cisco IP phones. This solution enables the large portfolio of Cisco access routers to deliver telephony features that are commonly used by business users to meet the requirements of the small or medium sized office. CallManager Express enables the deployment of a cost-effective, highly reliable, IP Communications solution using a single Cisco access router.

Customers can deploy Cisco CallManager at larger sites and deploy Cisco CallManager Express at branch office locations where local call processing is required. Using H.323, trunking calls can be routed over the wide area network with calling party name and number. A maximum of 240 IP phones can be supported (Cisco 3845 Integrated Services Router supports the maximum number) across a choice of platforms with CallManager Express.

We have used CallManager express (CME) on each of the client locations (routers MINT-EDM1 and MINT-CAL). To configure these routers with voice functionality using the CallManager express, the following steps need to be executed:

- Setting up DHCP service for Cisco CME
- Setting IP phones with CME
- Setting up initial extensions and Phones

[A] Setting up DHCP service for Cisco CME

When a Cisco IP phone is connected to the Cisco CME system, it automatically queries for a Dynamic Host Configuration Protocol (DHCP) server. The DHCP server responds by assigning an IP address to the Cisco IP phone and providing the IP address of the TFTP server through DHCP option 150. Then the phone registers with the Cisco CME server and attempts to get configuration and phone firmware files from the TFTP server. The configuration steps are listed in the following tabular display:

Command	Purpose
1. Router(config)# ip dhcp pool pool-name	Creates a name for the DHCP server address pool
2. Router(config-dhcp)# network <i>ip-address</i> [<i>mask</i> / <i>prefix-length</i>]	Specifies the IP address of the DHCP address pool.
3. Router(config-dhcp)# option 150 ip ip-address	Specifies the TFTP server address from which the Cisco IP phone downloads the image configuration file.
4. Router(config-dhcp)# default-router ip-address	Specifies the router that the IP phones will use to send or receive IP traffic
5. Router(config-dhcp)# exit	Exits DHCP pool configuration mode.

[B] Setting IP phones with CME

In a Cisco CME system, the IP phones receive their initial configuration information and phone firmware from the TFTP server associated with the Cisco CME router. In most cases, the phones obtain the IP address of their TFTP server using the Dynamic Host Configuration Protocol (DHCP) option 150 command. In other words, the IP-phone always looks for option 150 for the address of the TFTP server. In MINT-EDM1, this is configured as:

```

ip dhcp pool MINT
network 100.4.0.0 255.255.0.0
default-router 100.4.0.1
option 150 ip 100.4.0.1
domain-name Head-Office

```

For Cisco CME operation, the TFTP server address obtained by the Cisco IP phones should point to the Cisco CME router IP address. In our case it is the MINT-EDM1 router itself. The Cisco IP phones attempt to transfer a configuration file called XmlDefault.cnf.xml. This file is automatically generated by the Cisco CME router through the *ip source-address* command and placed in router memory. The XmlDefault.cnf.xml file contains the IP address that the phones use to register for service, using the Skinny Client Control Protocol (SCCP). This IP address should correspond to a valid Cisco CME router IP address (and may be the same as the router TFTP server address).

```

telephony-service
load 7960-7940 P00303020209
max-ephones 4
max-dn 4
ip source-address 100.4.0.1 port 2000
create cnf-files version-stamp Jan 01 2002 00:00:00
keepalive 45

```

The list of all the commands used in this context is available in the table below:

Command	Purpose
1. Router(config)# tftp-server flash:filename	Permits the Cisco CME router to provide TFTP access to the specified file by the IP phones served by the router
2. Router(config)# telephony-service	Enters telephony-service configuration mode
3. Router(config-telephony)# max-ephones max-phones	Sets the maximum number of Cisco IP phones to be supported by this router. Maximum – 48 In our case.
4. Router(config-telephony)# max-dn max-directory-numbers	Sets the maximum number of extensions (ephone-dns) to be supported by this router. Maximum – 192 in our case.
5. Router(config-telephony)# load <i>phone-type firmware-file</i>	Identifies the Cisco IP phone firmware file to be used by phones of the specified Cisco IP phone type when they register.
6. Router(config-telephony)# ip source-address <i>ip-address</i> [port port	Identifies the IP address and port number that the Cisco CME router uses for IP phone registration. The default port is 2000.

7. Router(config-telephony)# create cnf-files	Builds the XML configuration files that are required for Cisco CME phones.
8. Router(config-telephony)# keepalive <i>seconds</i>	Sets the time interval, in seconds, between keepalive messages that are sent to the router by Cisco IP phones. Default is 30.
9. Router(config-telephony)# reset all	Performs a fast reboot on one or all IP phones associated with the Cisco CME router.
9. Router(config-telephony)# exit	Exits telephony-service configuration mode.

[C] Setting up initial extensions

There are three ways to create an initial phone setup in a Cisco CME system:

- Automated Phone Setup Using the Cisco CME Setup Tool
- Partially Automated Setup Using the Router CLI
- Manual Setup Using the Router CLI

We have, however, used the manual setup option using router CLI. The manual process involves defining an *ephone-dn* (for setting up an extension) and an *ephone* (for setting up a Cisco CME phone). These terms are defined as follows:

- Ephone-dn – A software construct that represents a line that connects a voice channel to a phone instrument where a user can receive and make calls. An ephone-dn represents a virtual voice port in the Cisco CME system, so the maximum number of ephone-dns in a Cisco CME system is the maximum number of simultaneous call connections that can occur.
- Ephone – A software construct that represents a physical telephone instrument. The maximum number of ephones in a Cisco CME system is the maximum number of physical instruments that can be connected to the system

The command reference for setting up these parameters in the router, is available below:

Command	Purpose
1. Router(config)# ephone-dn <i>dn-tag</i>	Enters ephone-dn configuration mode.
2. Router(config)#number number	Configures a valid extension number for this ephone-dn instance.
3. Router(config-ephone-dn)#name name	Associates a name with this ephone-dn instance. This name is used for caller-ID displays and in the local directory listings.
4. Router(config-ephone-dn)#exit	Exits ephone-dn configuration mode.
5. Router(config)#ephone phone-tag	Enters ephone configuration mode.

6. Router(config-ephone)# mac-address mac-address	Specifies the MAC address of the IP phone that is being configured.
7. Router(config-ephone)# type type	Specifies the type of phone.
8. Router(config-ephone)# button <i>button-number</i>	Associates a button number and line characteristics with an extension (ephone-dn).
9. Router(config-ephone)# keepalive seconds	Sets the length of the time interval between successive keepalive messages from the Cisco CallManager Express router to a particular IP phone.
10 . . Router(config-ephone)#exit	Exits ephone configuration mode.

3.8 Chapter 4 - Preview

The objective of this report is to study the various techniques of Traffic Shaping. Every traffic-shaping technique has to pick a queuing method to go with. As a matter of fact, in some cases the behavior of the technique depends on the choice of the queue. Now, since queuing is an important integral chunk of the traffic shaping theory, we had to enhance our knowledge about the various queuing techniques that are available to use. So, before starting off with Traffic Shaping, lets have a brief tour of the prominent queuing technologies; especially the ones that we will be using in our experiments. Thus the following chapter is a discussion on the following queuing mechanisms:

- First-In-First-Out Queuing
- Priority Queuing
- Custom Queuing
- Weighted Fair Queuing
- IP RTP Prioritization

.....

CHAPTER 4

Queuing Mechanisms

4.1 Definition

Queuing on routers is necessary to accommodate bursts when the arrival rate of packets is greater than the departure rate due to one of the following two reasons:

- Input interface is faster than the output interface
- Output interface is receiving packets coming in from multiple other interfaces

Initial implementations of queuing used a single FIFO (first-in first-out or first-come first-serve queuing) strategy. More complex queuing mechanisms were introduced when special requirements need routers to differentiate between packets of different importance.

Queuing was split into two parts:

- The **hardware queue** that still uses FIFO strategy, which is necessary for the interface drivers to transmit packets one by one. The hardware queue is sometimes referred to as the *transmit queue* or TxQ.
- The **software queue** that schedules packets into the hardware queue based on the QoS requirements

Figure 4.1 illustrates the actions that have to be taken before a packet can be transmitted:

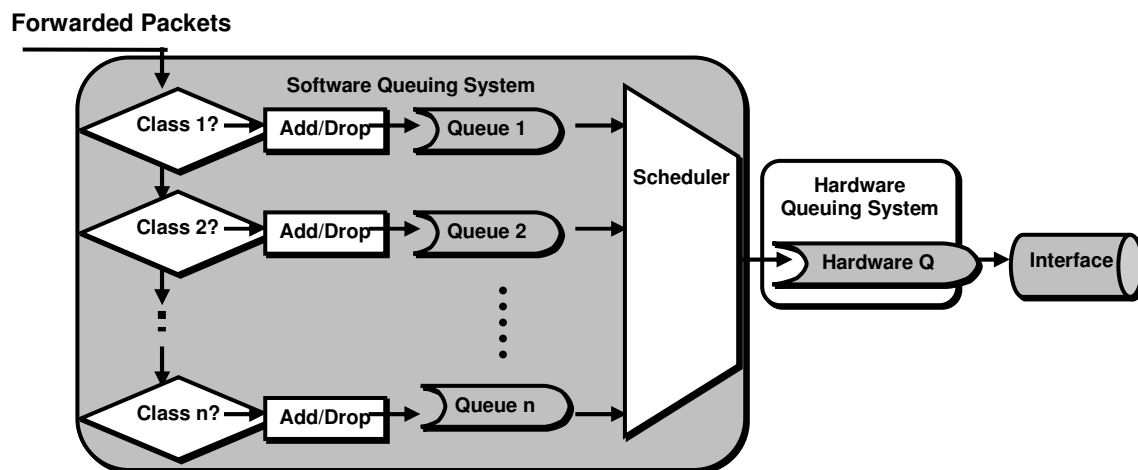


Figure 4.1

As is apparent from the diagram itself, each queuing mechanism has three main components that define it:

- Classification – selecting the class. Some mechanisms classify packets automatically (for example, WFQ), while other mechanisms require manual configuration of classification (for example, PQ or CQ).
- Insertion policy – determining whether a packet can be queued
- Service policy – scheduling packets to be put into the hardware queue. This is the most important part of every queuing mechanism because it determines the order in which the packets will leave the router.

The queuing mechanisms supported by the Cisco IOS are:

- FIFO: no differentiation of packets (true fairness but no guarantees)
- Priority Queuing (PQ): strict prioritizing of packets
- Custom Queuing (CQ): service (bandwidth) guaranteed to up to 16 classes
- Weighted Fair Queuing (WFQ) and Distributed WFQ: service (bandwidth) guarantee to individual flows
- Distributed ToS-based WFQ: service (bandwidth) guaranteed to up to 4 classes
- Distributed QoS-group-based WFQ: service (bandwidth) guaranteed to up to 100 classes
- Modified Deficit Round-robin (MDRR): service (bandwidth) guaranteed to up to 8 classes; low-delay guarantee if Strict or Alternate Priority is used
- IP RTP Prioritization: low-delay guarantee

4.2 First-in-first-out Queuing

*F*IFO queuing has no classification because all packets belong to the same class. Packets are dropped when the output queue is full (tail-drop). The scheduler services packets in the order they arrived. Software FIFO queue is basically an extension of the hardware FIFO queue.

FIFO queuing is regarded as the fairest queuing mechanism and is still the most used queuing mechanism. The following diagram illustrates the simple structure of the FIFO queuing mechanism.

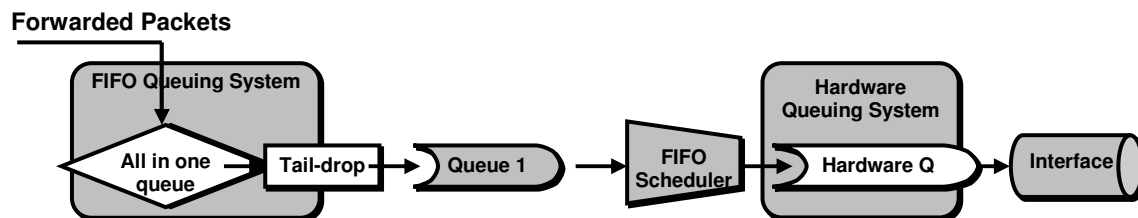


Figure 4.2

4.2.1 Features

*I*t is a simple and fast queuing system. Most high-end routers with fast interfaces are not really challenged by its limitations discussed in the next section. Furthermore, routers are not capable of complex classification and scheduling when they have to process a large number of packets per second. FIFO is, therefore, the most suitable queuing mechanisms on these platforms. It is supported on all platforms and all IOS versions.

4.2.2 Limitations

*F*IFO does **not fairly** allocate bandwidth among multiple flows. Some flows receive more bandwidth because they use larger packets or send more packets. The two most prominent drawbacks on FIFO are:

- FIFO is extremely unfair when an aggressive flow is contesting with a fragile flow. Aggressive flows send a large number of packets, many of which are dropped. Fragile flows send a modest amount of packets and most of them are dropped because the queue is always full due to the aggressive flow. This type of behavior is called **starvation**.
- Short or long bursts cause a FIFO queue to fill. Packets entering an almost full queue have to wait a long time before they can be transmitted. Another time, the queue might be empty causing packets of the same flow to experience almost no delay. Variation in delay is called **jitter**.

4.2.3 Configuring FIFO

*I*n Cisco routers, all interfaces with the default bandwidth above 2Mbps use FIFO queuing. No configuration is necessary on such interfaces.

All interfaces with the default bandwidth below 2Mbps use Weighted Fair Queuing (WFQ). The following command must be used to disable WFQ and enable FIFO.

```
config# no fair-queue
```

The following command can be used to increase or decrease the maximum number of buffered packets (applied at the interface configuration mode):

```
config-if# hold-queue buffer-size out
```

4.3 Priority Queuing

Priority Queuing (PQ) is one of the first mechanisms that allowed classification of packets into multiple classes. Scheduling is done in strict priority. PQ can classify packets into one of the four queues:

- High queue
- Medium queue
- Normal queue (the default queue)
- Low queue

Scheduling prefers packets in the same order. Each class uses one FIFO queue, where packets are dropped if a queue is full. The figure below illustrates the behavior of this queuing mechanism.

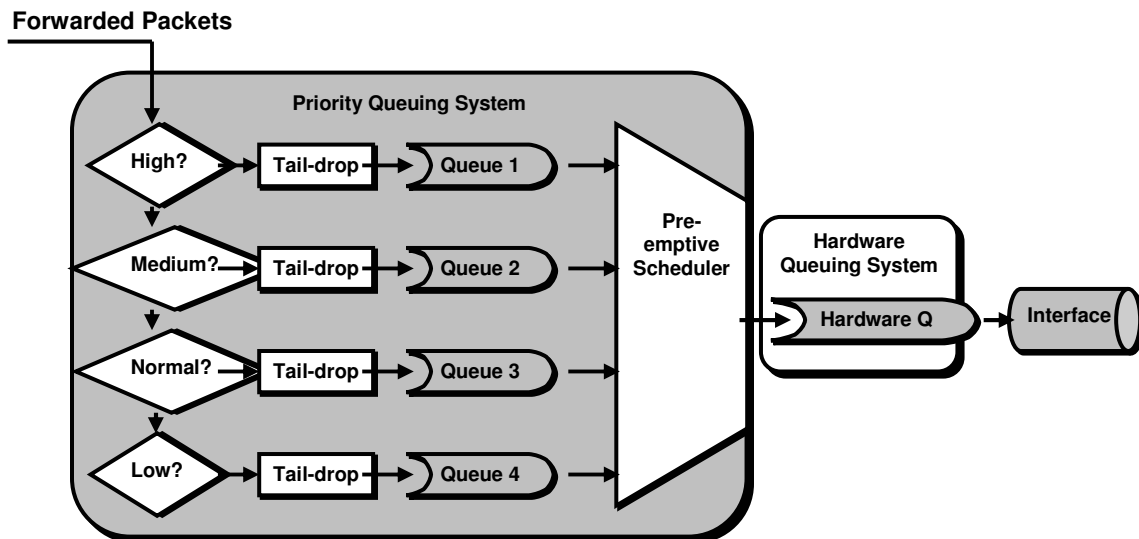


Figure 4.3

Priority Queuing can classify IP packets with the following tools:

- Direct matching on the source interface.
- Standard or extended IP Access list. Extended; matching on the parameters like:
 - Source or Destination IP address
 - Source or Destination TCP or UDP port number or port range
 - IP precedence (high-order three bits of the ToS field)
 - DSCP (high-order six bits of the ToS field)
 - ToS value (bits one through four of the ToS field)
 - TCP flags (ACK, SYN, RST, URG, PSH)
- Direct matching of TCP or UDP source and destination port numbers.
- Direct matching of fragments.
- Direct matching of packets based on their size.

4.3.1 Features

Priority Queuing is a multi-protocol QoS mechanism because it supports classification tools for other (non-IP) protocols. Some of the supported protocols are:

- IPX
- CLNS
- DECNET
- AppleTalk
- VINES

Priority Queuing is basically a collection of four parallel FIFO queues. Each queue uses the tail-drop scheme when the queue is full. It uses strict priority scheduling. As long as there are packets in the high queue no other queue will be served (refer figure 4.4). If the high queue is empty the router starts serving the medium queue.

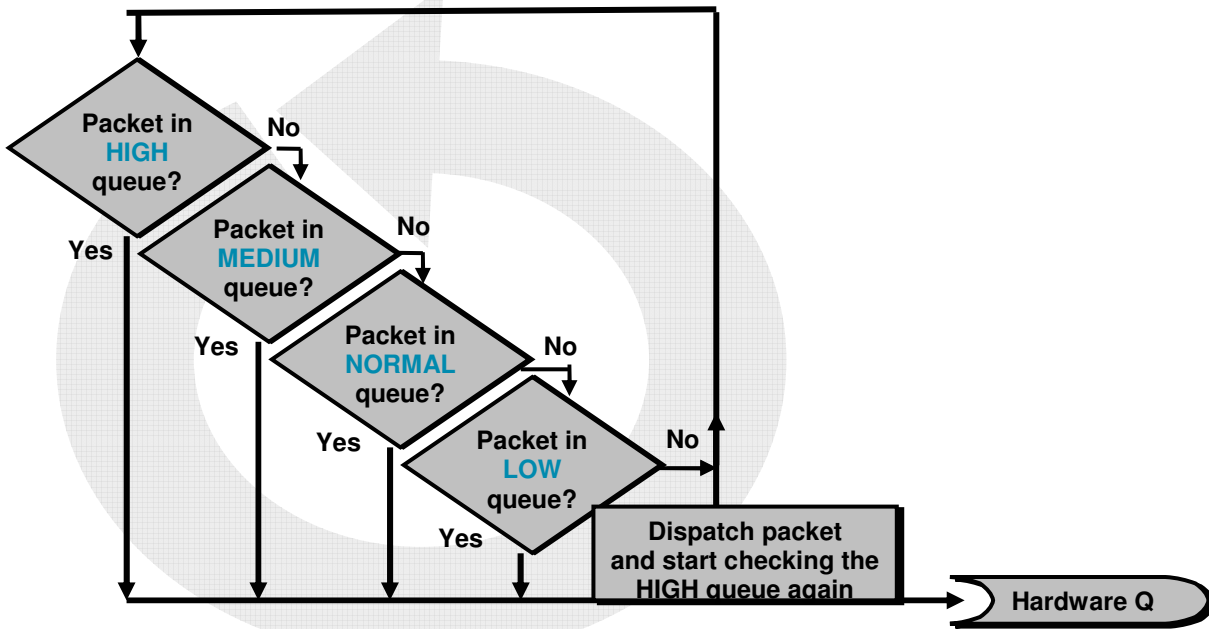


Figure 4.4

4.3.2 Limitations

Following are some of the prominent limitations of PQ:

- All limitations of FIFO queuing within a single class
- Starvation of lower -priority classes when higher priority classes are congested
- Manual configuration of classification on every hop

4.3.3 Configuring PQ

The configuration of Priority Queuing can be split into the following four steps:

- Classify data into four classes

To classify the packet based on incoming interface

```
priority-list list-number interface intf {high | medium | normal | low }
```

To classify all unclassified packets in a default queue

```
priority-list list-number default{high | medium | normal | low }
```

- Assign a queue to each class
- Set the maximum queue size (if the default is not appropriate)

```
priority-list list-number queue-limit high medium normal low
```
- Apply the priority queuing system to one or more interfaces

```
Priority-group list-number
```

4.4 Custom Queuing

Custom Queuing (CQ) is similar to Priority Queuing in the way it is configured and in the supported classification options. The scheduling, however, is completely different. The scheduling mechanism uses the round-robin service where each queue is allowed to forward a certain number of bytes (not packets). This is shown in Figure 4.5 in the next page.

CQ uses up to 16 queues that can be used for user-defined classes. The classification options are identical to those of Priority Queuing. Tail-drop is still used within each individual queue. Just like PQ, Custom Queuing can classify IP packets with the following tools:

- Direct matching on the source interface.
- Standard or extended IP Access list. Extended; matching on the parameters like:
 - Source or Destination IP address
 - Source or Destination TCP or UDP port number or port range
 - IP precedence (high-order three bits of the ToS field)
 - DSCP (high-order six bits of the ToS field)
 - ToS value (bits one through four of the ToS field)
 - TCP flags (ACK, SYN, RST, URG, PSH)
- Direct matching of TCP or UDP source and destination port numbers.
- Direct matching of fragments.
- Direct matching of packets based on their size.

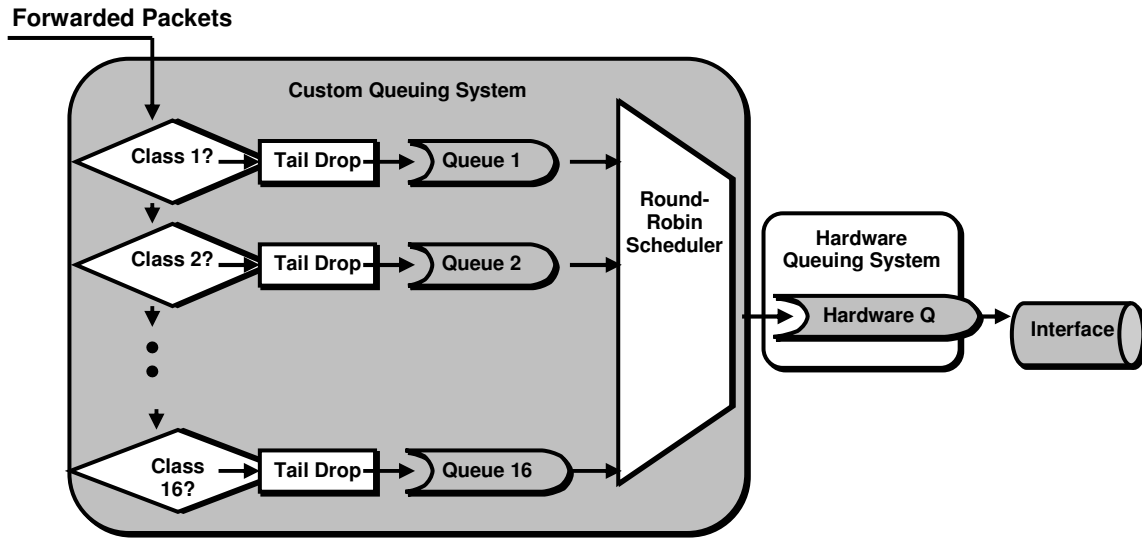


Figure 4.5

4.4.1 Features

Custom Queuing uses **round-robin** scheduling, where each queue gets some service (bandwidth). Each queue is configured with the number of bytes (byte-count) it can send in one round. The last packet is always sent, even if the total amount of bytes sent in one round is above the limit

Each queue, unless configured otherwise, can buffer up to 20 packets before the first packet is dropped. CQ guarantees throughput to traffic classes thus preventing starvation between them. It is supported on most platforms and in all IOS versions (above 10.0)

4.4.2 Limitations

The following are some of the limitations of CQ:

- All limitations of FIFO queuing within a single class
- Manual configuration of classification on every hop
- Not accurate bandwidth allocation
- High jitter due to implementation of scheduling

4.4.3 Configuring CQ

The configuration of Custom Queuing can be split into the following four steps:

- Classify the packets into various queues

To classify the packet into a custom queue based on incoming interface
queue-list *list-number* interface *incoming-intf* queue-number

To classify all unclassified packets into a default queue
queue-list *list-number* default queue-number

- Specify the scheduling parameters

To specify the lower boundary on how many bytes the system allows to be delivered from a given queue during one round-robin cycle
queue-list *list-number* queue *queue-number* byte-count *bc*

To specify the maximum number of packets in a queue
queue-list *list-number* queue *queue-number* limit *limit*

- Start custom queuing on an interface and assigns specified CQ definition to the interface

custom-queue *list-number*

4.5 Weighted Fair Queuing

Weighted Fair Queuing (WFQ) was introduced as a solution to the problems existing in the following queuing mechanisms:

- FIFO queuing causes starvation, delay and jitter
- PQ causes starvation of other lower-priority classes and suffers from all FIFO problems within each of the four queues
- CQ causes long delays and also suffers from all FIFO problems within each of the 16 queues

The idea of WFQ is to:

- Have a dedicated queue for each flow (no starvation, delay or jitter within the queue)
- Fairly and accurately allocate bandwidth among all flows (minimum scheduling delay, guaranteed service)
- Use IP precedence as weight when allocating bandwidth

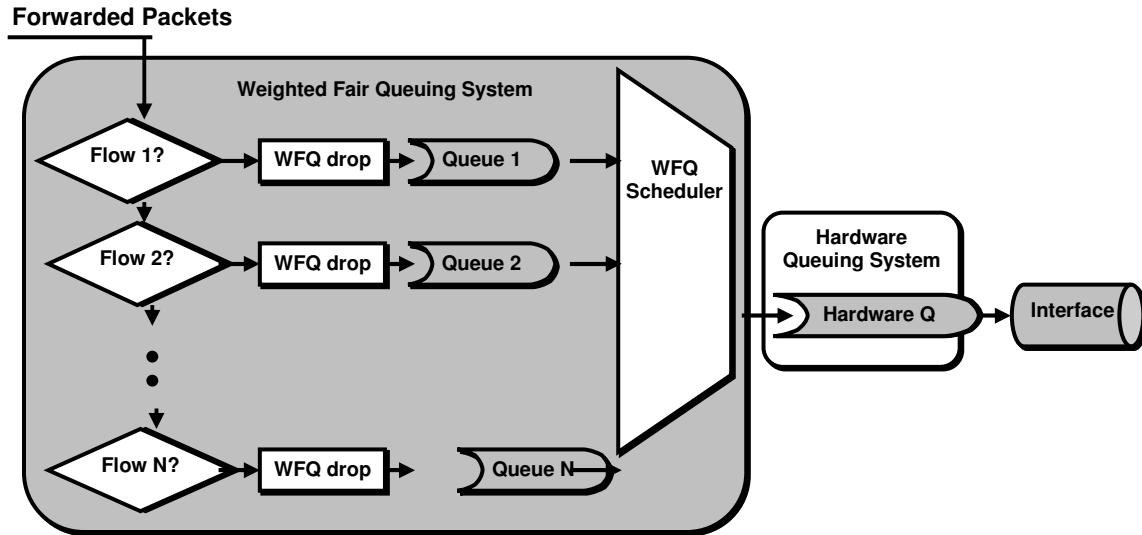


Figure 4.6

WFQ uses 256 queues by default. The number of queues can be configured in the range between 16 and 4096 (the number must be a power of 2). If there are a large number of concurrent flows it is very likely that two flows could end up in the same queue. It is recommended to have several times as many queues as there are flows (on the average).

Just as PQ and CQ, Weighted Fair Queuing can classify IP packets with the following tools as well:

- Direct matching on the source interface.
- Standard or extended IP Access list. Extended; matching on the parameters like:
 - Source or Destination IP address
 - Source or Destination TCP or UDP port number or port range
 - IP precedence (high-order three bits of the ToS field)
 - DSCP (high-order six bits of the ToS field)
 - ToS value (bits one through four of the ToS field)
 - TCP flags (ACK, SYN, RST, URG, PSH)
- Direct matching of TCP or UDP source and destination port numbers.
- Direct matching of fragments
- Direct matching of packets based on their size

4.5.1 Features

WFQ has two modes of dropping: Early dropping when the congestion discard threshold (CDT) is reached and Aggressive dropping when the hold-queue limit (HQO) is reached. WFQ always drops packets of the most aggressive flow

The length of queues (for scheduling purposes) is not in packets but in the time it would take to transmit all the packets in the queue. The end result is that WFQ adapts to the number of

active flows (queues) and allocates equal amounts of bandwidth to each flow (queue). The side effect is that flows with small packets (usually interactive flows) get a much better service because they do not need a lot of bandwidth. They, however, need low-delay, which they get because small packets have a low finish time.

WFQ uses the following IP precedence (According to the new formula) for calculating the weight of a flow in Cisco IOS versions after 12.0(5)T:

- IP precedence 0 weight 32384
- IP precedence 1 weight 16192
- IP precedence 2 weight 10794
- IP precedence 3 weight 8096
- IP precedence 4 weight 6476
- IP precedence 5 weight 5397
- IP precedence 6 weight 4626
- IP precedence 7 weight 4048

4.5.2 Limitations

The following are some of the limitations of CQ:

- All limitations of FIFO queuing within a single queue
- Multiple flows can end up in one queue
- Does not support the configuration of classification
- Can not provide fixed bandwidth guarantees
- Performance limitations due to complex classification and scheduling mechanisms

4.5.3 Configuring WFQ

WFQ is automatically enabled on all interfaces that have a default bandwidth of less than 2 Mbps. We can specify the maximum number of packets that can be in all output queues on the interface at any time using the following command (the default is 1000).

```
Config-if# hold-queue max-limit out
```

We can use the following commands to display the detailed information about the WFQ system:

```
Router# show interface interface  
Router# show queue interface
```

4.6 IP RTP Prioritization

IP RTP Prioritization is an add-on to WFQ to support low-delay propagation of packets and is usually used for VOIP traffic. It can be used for UDP traffic only (with predictable port numbers). If the destination UDP port is within the configured range it is queued into the high priority queue. It also polices the high priority traffic to prevent starvation of other queues.

As shown in the following figure IP RTP Prioritization supports one high priority queue. Packets from this queue are scheduled ahead of other packets as long as they are within the configured rate. Excess packets are dropped. A token Bucket model is used to measure the arrival rate of packets into this queue.

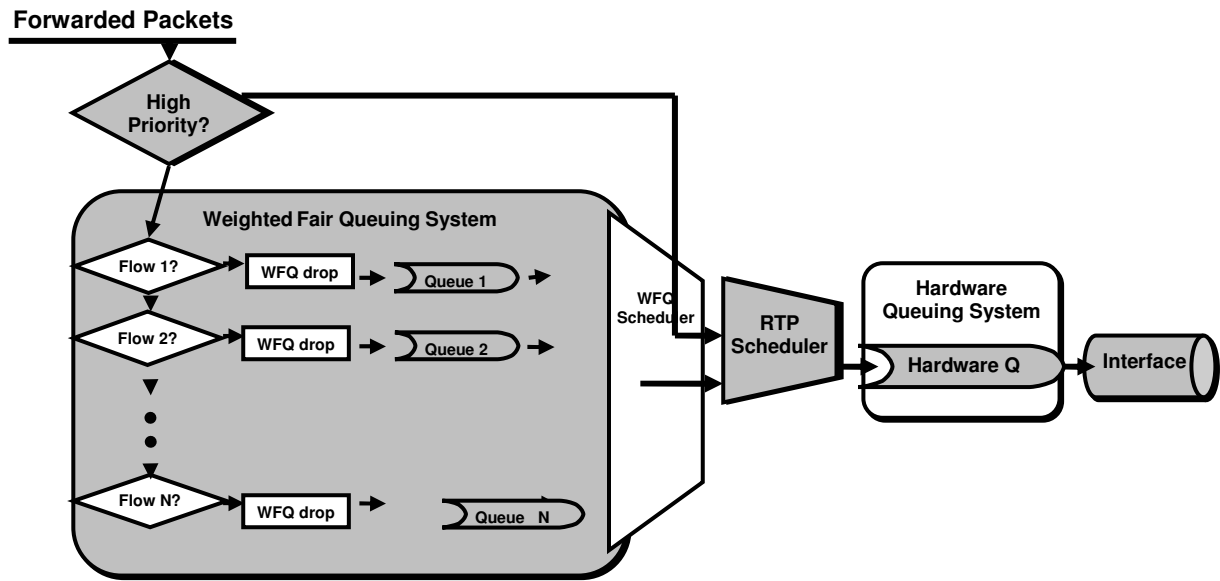


Figure 4.7

4.6.1 Features

The main benefiting features of IP RTP Prioritization is that it adds low-latency queuing to WFQ and CBWFQ and Prevents starvation of other traffic as well. Maximum bandwidth allocated to a queue is always slightly larger than actually required bandwidth due to jitter in the network and the Layer-2 overhead

4.6.2 Limitations

The main limitation of IP RTP Prioritization is that it has limited classification capabilities (UDP port range only). It was made obsolete by the introduction of Class-based Low Latency Queuing

4.6.3 Configuring IP RTP Prioritization

To create a separate priority queue for VoIP packets and specify maximum bandwidth available to voice traffic:

```
Router(config-if)# ip rtp priority starting-port port-range bandwidth
```

To reserve the maximum bandwidth percentage that can be allocated to class-based WFQ and priority RTP traffic:

```
Router(config-if)# max-reserved-bandwidth percent //default is 75%
```

4.7 Chapter 5 – Preview

The following chapters will be dealing with the different Traffic Shaping Mechanisms and their implementation. To start off, Chapter 5 deals with Generic traffic shaping concepts. GTS defines the foundation for all other techniques used for Traffic Shaping. It shapes traffic by reducing the outbound traffic flow to avoid congestion. The chapter starts with a discussion on GTS concepts and implementation procedures (command reference). We then move on to the practical implementation of the technology on our Lab network model; analyzing the results and the behavior of the shaping queues.

.....

CHAPTER 5

Generic Traffic Shaping

5.1 The Concept

Generic Traffic Shaping (GTS) defines the foundation for all other techniques used for Traffic Shaping. It shapes traffic by reducing the outbound traffic flow to avoid congestion. This is achieved by constraining traffic to a particular bit rate using the token bucket mechanism (discussed in Chapter-2 of this report). Figure 5.1 is a pictorial representation of the process involved in implementation of this mechanism. As is apparent from the figure, GTS performs three basic functions:

- Classification of traffic, so that different traffic classes can have different policies applied to them
- Metering, using a token-bucket mechanism, to distinguish between conforming and exceeding traffic
- Shaping, using buffering, to delay exceeding traffic and shape it to the configured rate limit

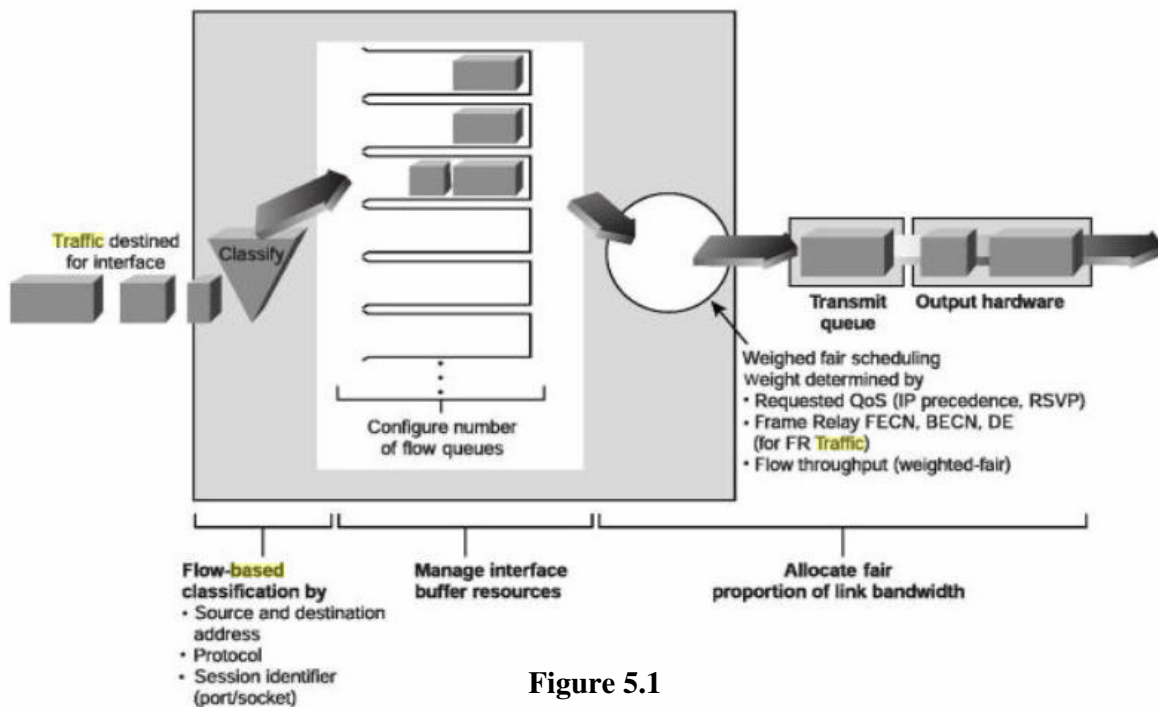


Figure 5.1

GTS is applied on a per-interface basis and can use access lists to select the traffic to shape. It works with a variety of Layer-2 technologies, including Frame Relay, ATM, Switched Multi-megabit Data Service (SMDS) and Ethernet. GTS is supported on most media and encapsulation types on the router. GTS can also be applied to a specific access list on an interface.

On a Frame Relay sub-interface, GTS can be set up to adapt dynamically to available bandwidth by integrating backward explicit congestion notification (BECN) signals, or set up simply to shape to a specified rate. GTS can also be configured on an ATM/ATM Interface Processor (AIP) interface to respond to the Resource Reservation Protocol (RSVP) feature signaled over statically configured ATM permanent virtual circuits (PVCs).

5.1.1 Queuing

GTS is implemented as a queuing mechanism, where there are separate WFQ delay queues implemented for each traffic class as shown in figure 5.2. Apparently, WFQ is the only queuing technique supported by GTS. Each WFQ-queue delays packets until they conform to the rate-limit, and also schedules them according to the WFQ algorithm. Conforming traffic is then sent to the physical interface. Arriving packets are first classified into one of the shaping classes. Classification can be performed using access lists.

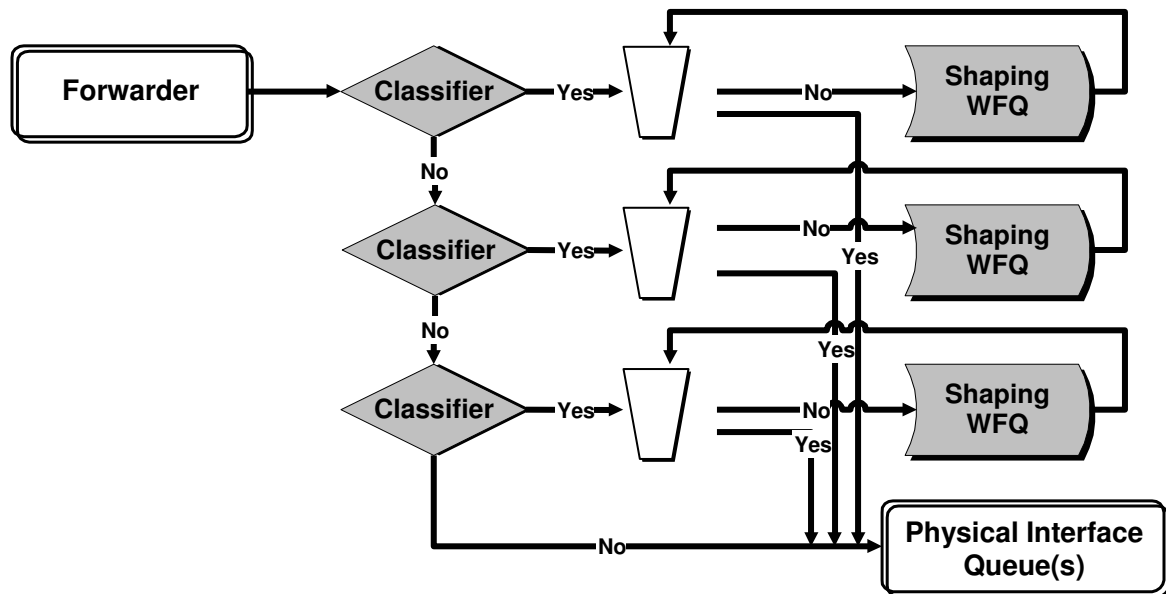


Figure 5.2

Once a packet is classified into a shaping class, its size is compared to the amount of available token in the token bucket of that class. The packet is forwarded to the main interface queue if there are enough tokens. A number of tokens taken out of the token bucket is equal to the size of the packet (in bytes).

If, on the other hand, there are not enough tokens to forward the packet, the packet is buffered in the WFQ system assigned to this shaping class. The router will then periodically replenish the token bucket and check if there are enough tokens to forward one or more packets out of the shaping queue. Packets are scheduled out of the shaping queue according to the WFQ scheduling algorithm.

5.1.2 GTS Features

*G*T*S* can be used to shape all outbound traffic on an interface or it can separately shape multiple classes. Classification is performed using any type of access list.

- GTS applies parameters per sub-interface
- GTS is multi-protocol
- GTS supports traffic group command
- Pure GTS can use only WFQ.
- GTS can be implemented in any queuing mechanisms:
 - FIFO Queuing
 - Priority Queuing (PQ)
 - Custom Queuing (CQ)
 - Weighted Fair Queuing (WFQ)
- GTS works on output only.
- GTS works on variety of interface types

Although different shaping techniques can be used with GTS, but in its pure implementation, it will only use WFQ. Application of other queuing techniques will change it to a different shaping technique altogether; for instance, GTS over CBWFQ will be called as Class-Based traffic shaping. In this chapter we will implement GTS with WFQ and a single queue. The class-concept will be exploited in Class Based Traffic Shaping (Chapter 6).

5.1.3 Restrictions

*F*ollowing are some of the prominent restrictions of GTS:

- GTS is not supported on the following interfaces:
 - Multilink PPP (MLP) interfaces
 - Integrated Services Digital Networks (ISDNs), dialer interfaces, or generic routing encapsulation (GRE) tunnel interfaces on the Cisco 7500 series router
- GTS is not supported with flow switching.

As discussed before, GTS is the basis for all the other traffic shaping techniques. It can be used with different queuing method to enhance its flexibility and thus rule out a lot of limitations.

5.2 Configuring GTS

Broadly speaking, GTS is supported on almost all the platforms except Integrated Services Digital Networks (ISDNs), dialup interfaces, or generic routing encapsulation (GRE) tunnel interfaces on the Cisco 7500 series router. (Also keeping in mind that traffic shaping is not supported with flow switching)

There are three different procedures that can be adopted to implement GTS, viz.:

- Generic Traffic Shaping on an Interface
- Generic Traffic Shaping Using an Access Control List
- Adaptive Generic Traffic Shaping for Frame Relay Networks

5.2.1 Configuring GTS on an Interface

To configure GTS on an interface, complete the following steps.

1. enable
2. configure terminal
3. interface *type number*
4. traffic-shape rate *bit-rate* [*burst-size*] [*excess-burst-size*] [*buffer-limit*]
5. end
6. show traffic-shape [*interface-type interface-number*]
7. show traffic-shape statistics [*interface-type interface-number*]
8. exit

Step 4 Enables traffic shaping for outbound traffic on an interface based on the bit rate specified. Steps 6 and 7 are used to display the current traffic-shaping configuration and statistics, respectively.

5.2.2 Configuring GTS using an Access Control List

Access control lists filter network traffic by controlling whether routed packets are forwarded or blocked at the router interface. When configured with GTS, the router examines each packet to determine how to shape the traffic on the basis of the criteria you specified for the access control list. Access control list criteria could be the source address of the traffic, the destination address of the traffic, the upper-layer protocol, or other information.

To configure GTS for outbound traffic using an access control list (ACL), complete the following steps:

1. enable
2. configure terminal
3. access-list *access-list-number* {deny | permit} source [*source-wildcard*]
4. interface *type number*
5. traffic-shape group *access-list bit-rate* [*burst-size* [*excess-burst-size*]]
6. end
7. show traffic-shape [*interface-type interface-number*]
8. show traffic-shape statistics [*interface-type interface-number*]
9. exit

The access list is defined in step 3 that shapes traffic according to specified criteria. Then in the step 5, *traffic-shape group* command is used to associate the access-list to an interface which then enables traffic shaping based on that specific access list for outbound traffic on that interface. Steps 7 and 8 are used to display the current traffic-shaping configuration and statistics, respectively

5.2.3 Configuring Adaptive GTS for Frame Relay Networks

On a Frame Relay (sub)interface, GTS can adapt dynamically to available Frame Relay bandwidth by integrating BECN signals. The GTS bit rate is reduced when BECN packets are received in order to reduce the data flow through the congested Frame Relay network; and then the bit rate is gradually increased when the congestion is no longer present (no BECN packets are received anymore).

To configure adaptive GTS for Frame Relay networks, complete the following steps.

1. enable
2. configure terminal
3. interface *type number*
4. traffic-shape rate *bit-rate* [*burst-size*] [*excess-burst-size*] [*buffer-limit*]
5. traffic-shape adaptive *bit-rate*
6. traffic-shape fecn-adapt
7. end
8. show traffic-shape [*interface-type interface-number*]
9. show traffic-shape statistics [*interface-type interface-number*]
10. exit

Shaping is enabled on a frame relay interface in Step 4 and then, in Step 5, that interface is configured to estimate the available bandwidth when BECNs are received. Step 6 Configures reflection of forward explicit congestion notifications (FECNs) as BECNs. Finally, Steps 8 and 9 are used to display the current traffic-shaping configuration and statistics, respectively.

5.3 Simulation Study

In the simulation study, we are mainly interested in studying the change in behavior of the network after the implementation of the Generic Traffic Shaping. We will consider two cases that will be implemented using our previously discussed LAB model (Figure 3.1).

5.3.1 Experimental Setup

In our setup the ISPs (Telus and Shaw) want to control the amount of traffic injected to the client network. As a matter of fact, our concern will be that Telus uses the frame relay network to connect to the Edmonton location whereas Shaw uses an Ethernet link to connect to the Calgary location. We will be using GTS in both the cases but the implementation will be slightly different. The section of the model used in this exercise is shown below:

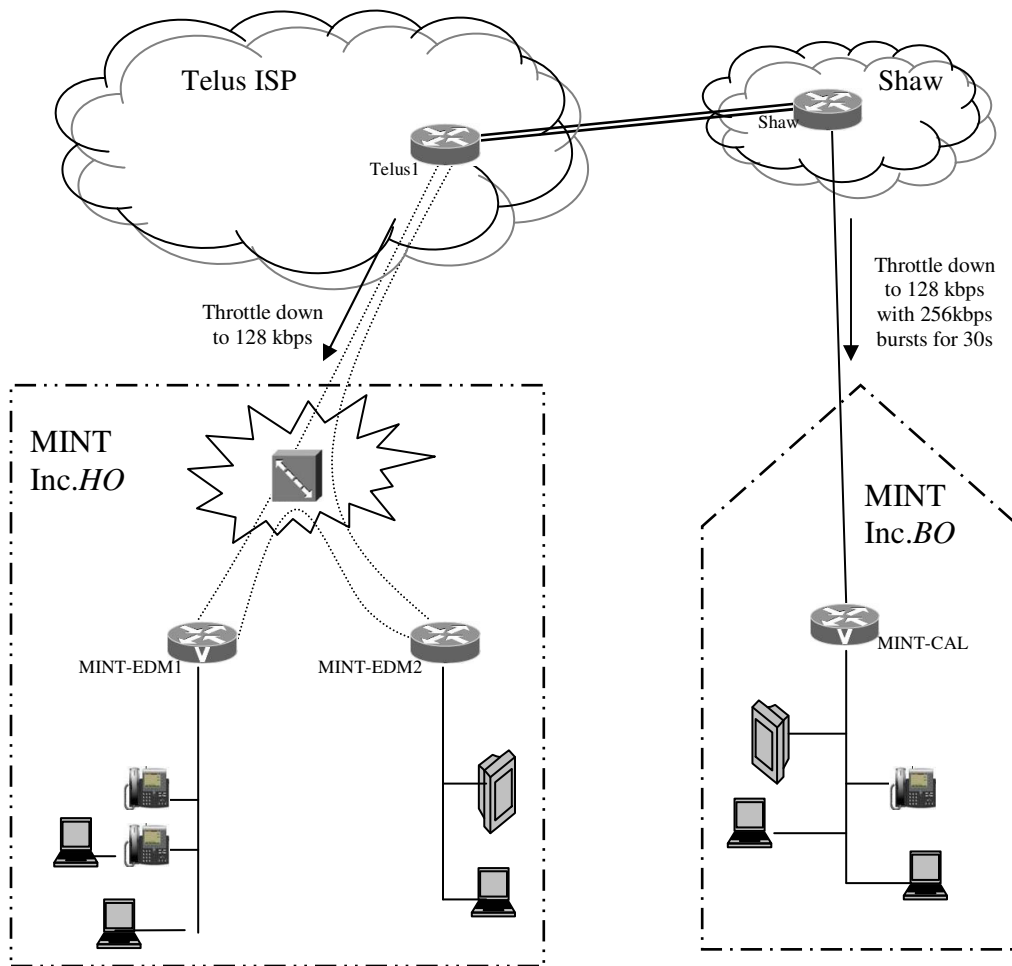


Figure 5.3

As Shown in the figure above, the limiting access rate for the client sides will be 128 Kbps. Additionally, Shaw also permits bursts up to of 256 Kbps for 30 seconds. Thus, the excess burst size will be $(256000 * 30) = 7680000$ bits. Now, since we are taking the default time interval of 125 msec, the burst size Bc can be calculated by the formula

$$\begin{aligned} Bc &= (CIR * Tc) \\ &= 128000 * 0.125 \\ &= 7936 \text{ Mbps} \end{aligned}$$

So, we can now list the values of the traffic shaping parameters:

$$\begin{aligned} CIR &= 128000 \\ Bc &= 7936 \text{ Mbps} \\ Be &= \text{zero for MINT-EDM1 and } 7680000 \text{ bits for Shaw} \\ Tc &= 125 \text{ msec} \end{aligned}$$

The routers are configured using the above values and commands referenced in section 5.2.2 of this chapter. The configuration done on each of these routers is shown below:

SHAW:

```
shaw#sh
Building configuration...

Current configuration : 1299 bytes
!
//output suppressed
!
interface GigabitEthernet0/1
 ip address 200.0.0.1 255.255.255.252
 duplex auto
 speed auto
 traffic-shape rate 12800 7936 7680000 1000
!
//output suppressed
!
end
shaw#
```

TELUS:

```
telus1#sh runn
Building configuration...

Current configuration : 2229 bytes
!
//output suppressed
!
interface Serial0/0.2 point-to-point
 ip address 100.1.10.1 255.255.255.252
 traffic-shape rate 12800 7936 0 1000
 frame-relay interface-dlci 20
!
//output suppressed
!
end
telus1#
```

Note that in case of the Telus1 router, shaping is implemented on the sub-interface serial0/0.2. Now, since our Frame Relay network is a point-to-point network and a single VC is configured per sub-interface, we are able to set the traffic shaping parameters for each VC just like we could have done using FRTS (Frame Relay Traffic Shaping). Had there been multiple VCs per sub-interface (point-to-multipoint), we would need FRTS to be able to set different parameters for each VC.

5.3.2 Results Analysis

We will now verify the implementation using the show commands and see how shaping queues are affected when video or voice data is passed through the traffic shaping channels. To view the shaping parameters configured on both the routers, use the “*show traffic-shape*” command in enable mode on each of them. The results are:

shaw# show traffic-shape

Interface	Access	Target	Byte	Sustain	Excess	Interval	Increment	Adapt
VC	List	Rate	Limit	bits/int	bits/int	(ms)	(bytes)	Active
-		12800	992	7936	7680000	661	992	-

telus1# show traffic-shape

Interface	Access	Target	Byte	Sustain	Excess	Interval	Increment	Adapt
VC	List	Rate	Limit	bits/int	bits/int	(ms)	(bytes)	Active
-		12800	992	7936	0	661	992	-

Now, we will play a video clip on the VLC video server located at the Head-office location (hosted by the MINT-EDM2 router) and stream it to the VLC client in the Branch office location (hosted by the MINT-CAL router). Since the serial link between the Telus1 and the MINT-CAL router was removed, the video stream will travel through the following path:

VLC Server → MINT-EDM2 → Telus1 → Shaw → MINT-CAL → VLC Client

This video clip was transcoded down to an average stream rate of 128 Kbps, which means that we are definitely expecting some concurrent bursts above and below that limit. Now since the channel is throttled to a rate of 128 Kbps by both the ISP routers, the bursts in the stream rate will fill the queues and thus trigger the traffic shaping mechanism. Our objective is to verify that the queues are filled when the data above the channel capacity is thrown over it and the traffic shaping mechanism smoothes the flow by delaying packets in the queue, preventing them to get dropped at the same time.

The “*show traffic-shape statistics*” command is used to display the current statistics of the shaping queues. So, when no video was played and the transmission rate was in the range on

the link bandwidth, the output of the above mentioned command shown no active traffic shaping and a queue length of zero.

shaw# show traffic-shape statistics

I/F	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
Gi0/1		0	170	15613	0	0	no

When video was played the queue starts filling up. The statistics will show the queue length at that instance, the total number of packets that have been delayed till then and the status of the shaping mechanism.

The following statistics for **Shaw** and **Telus1** were captured from recurrent execution of the “*show traffic-shape statistics*” command until the video was stopped. It is clearly apparent from the following data that shaping is triggered only when the queues are filled. As soon as the queues are emptied, shaping is turned off.

shaw# show traffic-shape statistics

I/F Gi0/1	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		64	269	111558	79	92827	yes
I/F Gi0/1	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		64	331	185025	141	166294	yes
I/F Gi0/1	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		55	352	211175	162	192444	yes
I/F Gi0/1	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		25	389	253012	199	234281	yes
I/F Gi0/1	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		15	546	453012	457	434281	yes
I/F Gi0/1	Acc. List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		0	985	948024	777	924968	no

telus1# show traffic-shape statistics

I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		64	97	108350	81	105169	yes
I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		40	163	194435	147	191254	yes

I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		29	174	209395	158	206214	yes
I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		17	189	225885	173	222704	yes
I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		7	199	239485	183	236304	yes
I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		1	205	247645	189	244464	yes
I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		0	206	249005	190	245824	no
I/F Se0/0.2	Access List	Queue Depth	Packets	Bytes	Packets Delayed	Bytes Delayed	Shaping Active
		0	206	249005	190	245824	no

Having proved this, we move on to the next level of the experiment. We will now transmit voice and video over the same channel and see the behavior of the queue as well as the shaping mechanism. For voice data, we have used the G.729 codec. It is an industry standard which allows for placing more calls in limited bandwidth to utilize IP voice in more cost effective ways. A typical call consumes 64Kbps of voice bandwidth whereas G.729 reduces the call to 8Kbps (normal IP overhead adds to this number). Practically speaking, the total bandwidth allocated to voice should be close to 30 Kbps which will include all the IP overheads as well.

Now, the queuing mechanism used in this experiment is the default Weighted Fair Queuing (because the link bandwidth is 128Kbps < 2Mbps) that prefers low bandwidth flows over high bandwidth flows. So, we will be sending voice (30 Kbps) and video (128 Kbps) over a link whose bandwidth is limited to 128 Kbps. Now, since we are using GTS, there is no mechanism defined to assign the bandwidth based on the type of traffic. By default, the delay sensitive voice traffic which requires low bandwidth is preferred over the video traffic that requires higher bandwidth. We will thus notice a noticeable drop of quality in the video and the video packets are delayed and dropped. At this point we will notice that despite of the low available bandwidth, there is no delay or packet drop in the voice traffic at all. The video traffic is compensated to save the voice traffic.

So, we played video at the VLC server (MINT-EDM2) streaming to the VLC client (MINT-CAL) and at the same time we started a VoIP call from Head-office location (MINT-EDM1) to the Branch-Office location (MINT-CAL). The queue depth in this case noticed a huge inflation due to the delayed video traffic, however there was no drop on the voice traffic at all. This can be verified using the “*debug ephone statistics mac address <mac-address>*” command at the voice enabled routers MINT-EDM1 and MINT-CAL. The statistics show

zero latency and zero packet loss in the voice data thus proving the preference of the voice data over the video data:

MINT-EDM1# debug ephone statistics mac address 0007.8505.631D

```
*Mar 2 20:25:48.556: ephone-1[1]:GetCallStats line 2 ref 16 DN 2: 1011
*Mar 2 20:25:48.808: ephone-1[1]:Call Stats for line 2 DN 2 1011 ref
16
*Mar 2 20:25:48.808: ephone-1[1]:TX Pkts 692 bytes 22144 RX Pkts 685
bytes 21891
*Mar 2 20:25:48.808: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
*Mar 2 20:25:48.808: ephone-1[1]:Src 100.4.0.6 22414 Dst 100.4.0.1
2000 bytes 20 vad 250 G729
*Mar 2 20:25:52.420: STATS: DN 2 Packets Sent 692
*Mar 2 20:25:52.420: STATS: DN 2 Packets Received 685
*Mar 2 20:25:52.420: STATS: DN 2 Latency 0
*Mar 2 20:25:52.420: STATS: DN 2 PacketsLost 0
*Mar 2 20:25:53.600: ephone-1[1]:GetCallStats line 2 ref 16 DN 2: 1011
*Mar 2 20:25:53.852: ephone-1[1]:Call Stats for line 2 DN 2 1011 ref
*Mar 2 20:25:53.852: ephone-1[1]:TX Pkts 945 bytes 30240 RX Pkts 938
bytes 29987
*Mar 2 20:25:53.852: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
*Mar 2 20:25:53.852: ephone-1[1]:Src 100.4.0.6 22414 Dst 100.4.0.1
2000 bytes 20 vad 250 G729
*Mar 2 20:25:57.376: STATS: DN 2 Packets Sent 945
*Mar 2 20:25:57.376: STATS: DN 2 Packets Received 938
*Mar 2 20:25:57.376: STATS: DN 2 Latency 0
*Mar 2 20:25:57.376: STATS: DN 2 PacketsLost 0
```

MINT-CAL# debug ephone statistics mac address 0007.5079.BF61

```
*Mar 5 02:02:19.063: ephone-1[1]:GetCallStats line 2 ref 20 DN 2: 2011
*Mar 5 02:02:19.315: ephone-1[1]:Call Stats for line 2 DN 2 2011 ref
20
*Mar 5 02:02:19.315: ephone-1[1]:TX Pkts 8915 bytes 285165 RX Pkts
6769 bytes 216275
*Mar 5 02:02:19.315: ephone-1[1]:Pkts lost 0 jitter 550 latency 0
*Mar 5 02:02:19.315: ephone-1[1]:Src 100.8.0.6 21406 Dst 100.8.0.1
2000 bytes 20 vad 250 G729
*Mar 5 02:02:23.339: STATS: DN 2 Packets Sent 8915
*Mar 5 02:02:23.339: STATS: DN 2 Packets Received 6769
*Mar 5 02:02:23.339: STATS: DN 2 Latency 0
*Mar 5 02:02:23.339: STATS: DN 2 PacketsLost 0
*Mar 5 02:02:24.107: ephone-1[1]:GetCallStats line 2 ref 20 DN 2: 2011
*Mar 5 02:02:24.359: ephone-1[1]:Call Stats for line 2 DN 2 2011 ref
*Mar 5 02:02:24.359: ephone-1[1]:TX Pkts 9167 bytes 293229 RX Pkts
6771 bytes 216301
*Mar 5 02:02:24.359: ephone-1[1]:Pkts lost 0 jitter 566 latency 0
*Mar 5 02:02:24.359: ephone-1[1]:Src 100.8.0.6 21406 Dst 100.8.0.1
2000 bytes 20 vad 250 G729
*Mar 5 02:02:26.867: STATS: DN 2 Packets Sent 9167
*Mar 5 02:02:26.867: STATS: DN 2 Packets Received 6771
*Mar 5 02:02:26.867: STATS: DN 2 Latency 0
*Mar 5 02:02:26.867: STATS: DN 2 PacketsLost 0
```

5.3.3 Conclusion

Generic traffic shaping works on an entire interface to limit the rate that it sends data. We can also specify traffic shaping for packets that match a particular access list. This will buffer only the matching traffic, and leave all other traffic to use the default queuing mechanism for the interface. GTS uses WFQ as the default queuing mechanism and thus it gives a default high priority to the voice traffic that requires low bandwidth (~30Kbps) over the video traffic whose bandwidth requirements reach up to 128Kbps.

The majority of the data traffic that we generated in our experiment is real time. This is usually carried with the UDP protocol. We have used the traffic sniffers like Ethereal to capture traffic on the active interfaces and thus determining that tests are being performed on the desired traffic to satisfy the objective of the experiment. As an example figure 5.4 and 5.5 below show the network activity at the Telus1 router during the streaming of the video from the video server at 154.0.0.2 to the video client at 100.8.0.7. That majority of the traffic is UDP.

```

Frame 1 (1370 bytes on wire (1370 bytes captured)
Arrival Time: Jul 27, 2007 16:15:43.030419000
[Time delta from previous packet: 0.000000000 seconds]
[Time since reference or first frame: 0.000000000 seconds]
Frame Number: 1
Packet Length: 1370 bytes
Capture Length: 1370 bytes
[Protocols in frame: eth:ip:udp:data]
Ethernet II, Src: 100.8.0.1 (00:08:21:96:f3:00), Dst: Dell_dc:f8:b4 (00:12:3f:dc:f8:b4)
Destination: Dell_dc:f8:b4 (00:12:3f:dc:f8:b4)
Source: 100.8.0.1 (00:08:21:96:f3:00)
Type: IP (0x0800)
Internet Protocol, Src: 154.0.0.2 (154.0.0.2), Dst: 100.8.0.7 (100.8.0.7)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
  0000 00.. = Differentiated Services Codepoint: Default (0x00)
  .... ..0. = ECN-Capable Transport (ECT): 0
  .... ..0. = ECN-CE: 0
Total Length: 1356
Identification: 0x297a (10618)
Flags: 0x00
Fragment offset: 0
Time to live: 126
Protocol: UDP (0x11)
Header checksum: 0x1016 [correct]
Source: 154.0.0.2 (154.0.0.2)
Destination: 100.8.0.7 (100.8.0.7)
User Datagram Protocol, Src Port: 1099 (1099), Dst Port: 1234 (1234)
Source port: 1099 (1099)
Destination port: 1234 (1234)
Length: 1336
Checksum: 0x777c [correct]
Data (1328 bytes)

0000  00 12 3f dc f8 b4 00 08 21 96 f3 00 08 00 45 00  ..?....!....E.
0010  05 4c 29 7a 00 00 7e 11 10 16 9a 00 00 02 64 08  .L)z... ..d.
0020  00 07 04 4b 04 d2 05 38 77 7c 80 21 42 86 94 05  ...K...8 w|.!B...
0030  57 b4 00 00 78 1b 47 40 0a 15 00 00 01 e0 02 c8  W...X.G@ .....
0040  80 c0 0a 3d 50 13 d3 67 1d 50 13 d3 67 00 00 01  ...=P..g .P..g...
0050  bc 7e ff ff ff ff ff ff ff ff ff ff ff ff ff  ..

Ethernet (eth), 14 bytes |P: 44 D: 44 M: 0
```

Figure 5.4

Thus the functionality of the GTS was tested and verified using a number of tools. The only prominent limitation of GTS in our scenario is that it is not supported at multilink PPP interfaces.

No. -	Time	Source	Destination	Protocol	Info
1	0.000000	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
2	0.663969	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
3	1.327995	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
4	2.644136	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
5	3.308008	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
6	4.628178	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
7	5.292047	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
8	6.608122	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
9	6.656679	Cisco_96:f3:00	Cisco_96:f3:00	LOOP	Reply
10	7.272080	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
11	7.877051	100.8.0.1	Broadcast	ARP	who has 100.8.0.6? Tell 100.8.0.1
12	7.936047	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
13	9.252194	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
14	9.916239	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
15	10.017483	100.8.0.7	100.8.255.255	BROWSE	Domain/workgroup Announcement WORKGROUP,
16	11.236195	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
17	11.900248	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
18	12.561236	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
19	13.880215	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
20	14.544317	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
21	15.864452	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
22	16.528301	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
23	16.657146	100.8.0.1	100.8.0.1	LOOP	Reply
24	17.573488	100.8.0.1	Broadcast	ARP	who has 100.8.0.6? Tell 100.8.0.1
25	17.848333	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
26	18.512244	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
27	19.176396	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
28	20.492432	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
29	21.156292	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
30	22.472357	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
31	23.136482	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
32	23.800405	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
33	25.116429	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
34	25.780493	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234
35	26.657529	100.8.0.1	100.8.0.1	LOOP	Reply
36	27.100505	154.0.0.2	100.8.0.7	UDP	Source port: 1099 Destination port: 1234

Figure 5.5

This means that if, in our scenario, we would have used multilink frame relay VCs at the ISP end, GTS would not have been an option to set the traffic shaping parameters at each of these VCs. This means that it can only shape traffic for a physical or logical interface but link level shaping is not supported by GTS. This explain as to why it is not supported at other networks like ISDN as well.

5.4 Chapter 6 - Preview

The next chapter deals with an improved mechanism for traffic shaping called Class Based Traffic Shaping. CBTS enables us to configure traffic shaping on a per-traffic-class basis rather that restricting it to per-flow basis. We can define different classes of traffic using in a hierarchical policy map structure thus accomplishing shaping at a more granular level. The chapter starts with a discussion on CBTS concepts and implementation procedures. We then move on to the practical implementation of the technology on our Lab network model; analyzing the results and the behavior of the shaping queues.

.....

CHAPTER 6

Class Based Traffic Shaping

6.1 The Concept

Packet flow on a network can be regulated using a traffic shaping mechanism. After having a detailed discussion on the Generic traffic shaping mechanism, let us talk about another such feature called Class-Based Traffic Shaping. Class-Based Traffic Shaping allows us to regulate the flow of packets (on a per-traffic-class basis) going out an interface, matching the packet flow to the speed of the interface. It holds and shapes traffic to a particular bit rate by using the token bucket mechanism (Chapter-2).

Using the Class-Based Traffic Shaping, we can perform the following tasks:

- Configure traffic shaping on a per-traffic-class basis. It allows us to fine-tune traffic shaping for one or more classes and it allows us to configure traffic shaping on a more granular level.
- Specify average rate or peak rate traffic shaping. Specifying peak rate shaping allows us to make better use of available bandwidth by allowing more data than the configured traffic shaping rate (CIR) to be sent if the bandwidth is available. To determine the peak rate, the router uses the following formula:

$$\text{Peak rate} = \text{CIR}(1 + \text{Be}/\text{Bc})$$

where:

- Be is the Excess Burst rate.
- Bc is the Committed Burst rate.

However, using peak rate shaping, the traffic sent above the CIR (the delta) has the potential of being dropped if the network becomes congested. If the network has additional bandwidth available (over the provisioned CIR) and the application or class can tolerate occasional packet loss, that extra bandwidth can be exploited through the use of peak rate shaping. However, there may be occasional packet drops when network congestion occurs. If the traffic being sent to the network must strictly conform to the configured network provisioned CIR, then we should use average traffic shaping.

- Configure traffic shaping in a hierarchical policy map structure. That is, traffic shaping is configured in a primary-level (parent) policy map and other QoS features (for instance, CBWFQ and traffic policing) can be configured in the secondary-level (child) policy maps.

6.1.1 Queuing

The two queuing mechanisms supported by Class based traffic shaping are:

- Weighted Fair Queuing (WFQ)
- Class Based Weighted Fair Queuing (CBWFQ).

Flow-based WFQ applies weights to traffic to classify it into conversations and determine how much bandwidth each conversation is allowed relative to other conversations. These weights, and traffic classification, are dependent on and limited to the seven IP Precedence levels discussed in Chapter 4.

CBWFQ is loosely based on WFQ and CQ. It allows us to define what constitutes a class based on criteria that exceed the confines of flow. CBWFQ allows the use of ACLs and protocols or input interface names to define how traffic will be classified, thereby providing coarser granularity. We need not maintain traffic classification on a flow basis. Moreover, we can configure up to 64 discrete classes in a service policy.

The limitation of flow-based WFQ is that the flows are automatically determined, and each flow gets a fair share of the bandwidth. This “fair share” of the bandwidth is determined by the size of the flow and moderated by IP precedence. Packets with IP precedence set to values other than the default (zero) are placed into queues that are serviced more frequently, based on the level of IP precedence, and thus get a higher overall bandwidth. Weight is a relative number and the lower the weight of a packet, the higher that packet’s priority. The most important concern is the packet’s specific handling. Thus, a data stream with a precedence of 1 is dealt with twice as fast as best-effort traffic. However, even with the action of IP Precedence on WFQ, sometimes a specific bandwidth needs to be guaranteed to a certain type of traffic. CBWFQ fulfills this requirement as the different traffic types can be filtered based on Protocols, ACLs or input interfaces and the weight is assigned based on the bandwidth allocated to each type of traffic (traffic class).

In our experiments we will implement CBWFQ as the queuing mechanism for the Class Based Traffic Shaping.

6.1.2 CBTS Features

CBTS can be used with Class Based Traffic Shaping that combines the features of Weighted Fair Queuing (WFQ) and Custom Queuing (CQ) and that is why it is also referred to as Hybrid Congestion Control Mechanism. The following features available with CBTS distinguish it from the other shaping mechanisms:

- Specifying Average Rate or Peak Rate Traffic Shaping – In addition to using a specifically configured transmission rate, we can use class based traffic shaping to specify a derived transmission rate based on the level of congestion. We can specify

two types of traffic shaping; *average rate* shaping and *peak rate* shaping. Average rate shaping limits the transmission rate to the committed information rate (CIR). Using the CIR ensures that the average amount of traffic being sent conforms to the rate expected by the network. On the contrary, Peak rate shaping configures the router to send more traffic than the CIR. Peak rate shaping allows the router to burst higher than average rate shaping.

- Hierarchical Policy Map Structure – With the Class-Based Traffic Shaping mechanism, traffic shaping can be configured in a hierarchical policy map structure; that is, traffic shaping is enabled in a primary-level (parent) policy map and other QoS features used with traffic shaping, such as CBWFQ and traffic policing, can be enabled in a secondary-level (child) policy map. CBWFQ, however, is supported in both the primary-level (parent) policy map and the secondary-level (child) policy map. But, to use CBWFQ at the secondary-level (child) policy map, traffic shaping must be configured in the primary-level (parent) policy map.
- Flexibility of Match Criteria – Applying traffic shaping to classes provides greater flexibility for configuring traffic shaping. Previously, this ability was limited to the use of ACLs.
- Coarser Granularity and Scalability – CBWFQ allows you to use ACLs and protocols or input interface names to define how traffic will be classified, thereby providing coarser granularity. You need not maintain traffic classification on a flow basis. We can configure up to 64 discrete classes in a service policy.

6.1.3 Restrictions

Some significant restrictions that defines that upper bounds for the capacity of Class Based Traffic Shaping are:

- Peak and average traffic shaping is configured on a per-interface or per-class basis, and cannot be used in conjunction with commands used to configure traffic shaping from previous versions of Cisco IOS. These commands include the following:
 - traffic-shape adaptive
 - traffic-shape fecn-adaptive
 - traffic-shape group
 - traffic-shape rate
- Adaptive traffic shaping for Frame Relay networks is not supported using the Class-Based Shaping feature.
- Class-Based Traffic Shaping applies to outbound traffic only.

6.2 Configuring CBTS

Implementation procedure of Class Based Traffic Shaping can vary depending of the policy-map structure. The presence of hierarchy in the policy-map will demand the definition of secondary-level (child) policy-maps within the primary level (parent) policy-maps. We thus define two distinct procedures involved in configuring CBTS:

- Configuring CBTS in a Primary-Level (Parent) Policy Map
- Configuring CBTS with Secondary-Level (Child) Policy Map

6.2.1 CBTS in a Primary-Level (Parent) Policy Map

To configure Class-Based Traffic Shaping (after first creating a policy map and class map), complete the following steps.

1. enable
2. configure terminal
3. policy-map *policy-map-name*
4. class {*class-name* | class-default}
5. shape [average | peak] *mean-rate* [[*burst-size*] [*excess-burst-size*]]
6. service-policy *policy-map-name*
7. end
8. show policy-map
9. show policy-map interface *interface-name*
10. exit

Step 3 defines a policy map and enters the policy-map configuration mode and Step 4 specifies a class whose policy needs to be defined and enters the policy-map class configuration mode. The *shape* command in Step 5 is used to enter the shaping parameters; we can also specify here, whether we want the traffic to be shaped on an *average* rate or the *peak* rate. Step 6 can be used to specify a service-policy as a QoS policy within the policy-map (called hierarchical service policy). Finally steps 8 and 9 are monitoring commands, step 8 is used to list all the configured policy-maps and step 9 displays the packet statistics of all classes that are configured for all service policies either on the specified interface or sub-interface or on specific PVC on the interface.

6.2.2 CBTS with Secondary-Level (Child) Policy Map

In the secondary-level (child) policy map, additional QoS features used with traffic shaping (for example, CBWFQ and traffic policing) are typically configured. For Class-Based Traffic

Shaping, the only two QoS features supported at the secondary-level (child) policy map are CBWFQ and traffic policing. To configure these QoS features in a secondary-level (child) policy map, complete the following steps

1. enable
2. configure terminal
3. policy-map *policy-map-name*
4. class {*class-name* | class-default }
5. bandwidth {*bandwidth-kbps* | remaining percent *percentage* | percent *percentage* }
6. end
7. show policy-map
8. show policy-map interface *interface-name*
9. exit

Step 3 defines a policy map and enters the policy-map configuration mode and Step 4 specifies a class whose policy needs to be defined and enters the policy-map class configuration mode. The *bandwidth* command in step 5 configures CBWFQ (we can use *police* to configure traffic policing). This command specifies or modifies the bandwidth allocated for a class belonging to a policy map. We can enter the amount of bandwidth as a number of kbps, a relative percentage of bandwidth, or an absolute amount of bandwidth. Finally steps 7 and 8 are monitoring commands, step 7 is used to list all the configured policy-maps and step 8 displays the packet statistics of all classes that are configured for all service policies either on the specified interface or sub-interface or on specific PVC on the interface.

6.3 Simulation Study

*I*n the simulation study, we are mainly interested in studying the change in behavior of the network after the implementation of the Class Based Traffic Shaping. We will consider two cases that will be implemented using our previously discussed LAB model (Figure 3.1).

6.3.1 Experimental Setup

*I*n this implementation, Telus has an agreement with MINT Inc. to support voice and video streaming services through the frame relay network to the branch location at Calgary. Now, we have a bandwidth limitation in the scenario that needs to be managed to support the above mentioned data types. We know that voice is a delay sensitive traffic and thus would need highest priority over other types of data. We are using the G.729 codec which would need, practically, 30 to 40 Kbps of channel capacity to be transmitted. On the other hand, video

bandwidth depends on the quality of the video being transmitted. As a matter of fact, video is not a delay-sensitive traffic and thus could be placed at a lower precedence to voice.

Thus, keeping in mind all the requirements and the reservations, the following design was formulated for shaping traffic: The video traffic can consume all the bandwidth, if there is no voice data on the channel. And the voice data will get a dedicated 50 percent of the total bandwidth at any instance when it needs to be transmitted.

Consider the network setup depicted in Figure 6.1. We will commit a guaranteed 50% bandwidth to voice on the peak rate and Video will get the 40% of the bandwidth but will be able to use the complete channel when voice is not being transmitted.

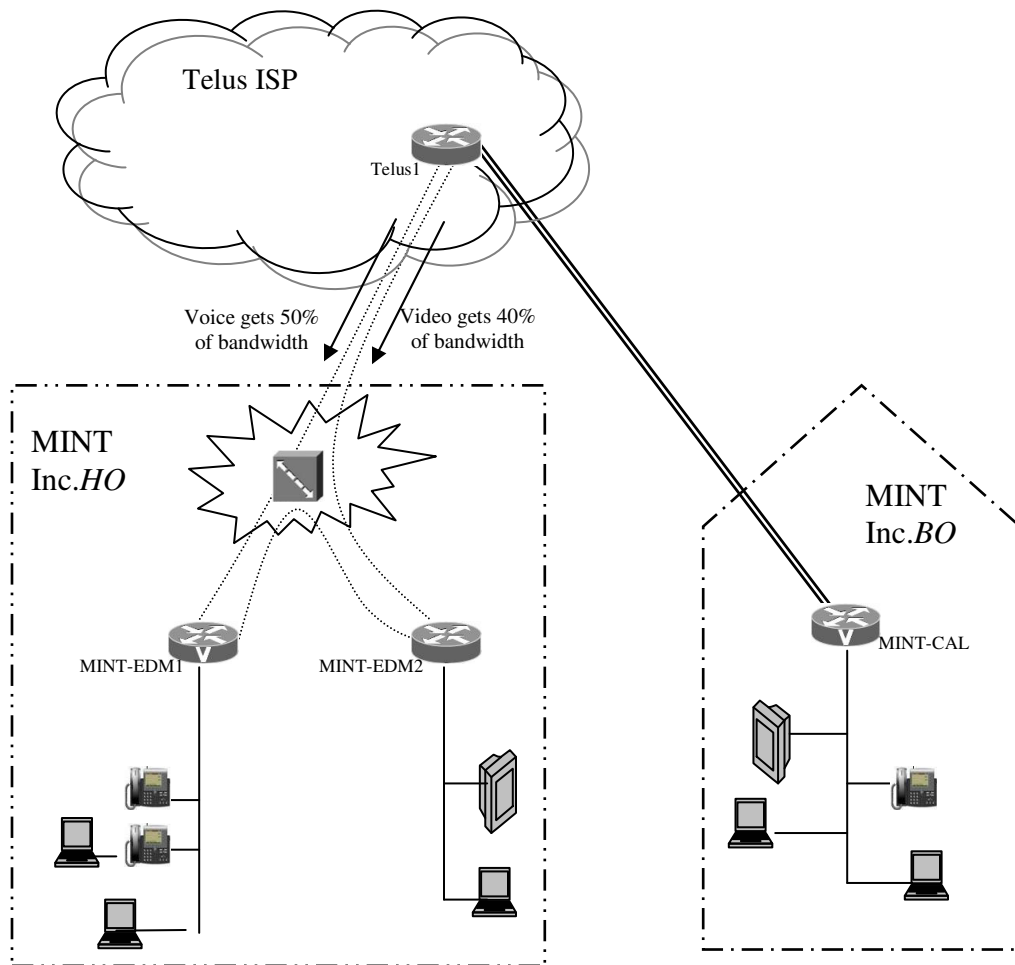


Figure 6.1

To implement this using Class Based Traffic Shaping, we have defined a Policy named “CBWFQ” and two traffic classes named “voice” and “data” are defined under this policy. All the data except *voice*, will be classified in the “data” class. This is represented pictorially in the following figure.

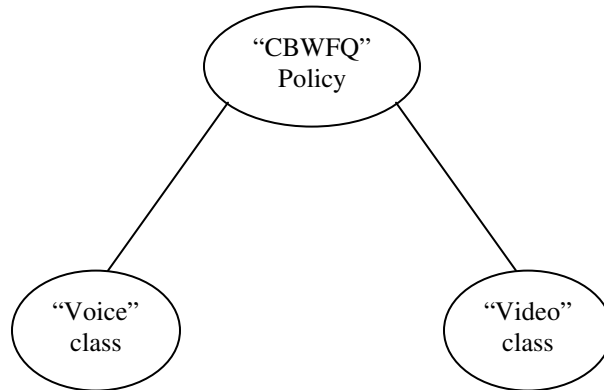


Figure 6.2

The routers are configured based on the above values and commands referenced in section 6.2.2 of this chapter. The configuration is mainly done on the ISP router, Telus1. A view of the configuration file is shown below

TELUS1 :

```

telus1#sh runn
Building configuration...

Current configuration : 2574 bytes
!
//output suppressed
!
!
!
class-map match-all data
  match input-interface Serial0/1
class-map match-any voice
  match ip rtp 16384 16383
!
!
policy-map CBWFQ

  class voice
    bandwidth percent 50
    queue-limit 10
    shape peak 128000
  class data
    bandwidth percent 40
    queue-limit 20
    shape average 12800
!
!
!
interface Serial0/0
  bandwidth 10000000
  no ip address
  service-policy output CBWFQ

```

```

encapsulation frame-relay
clock rate 128000
!
!
//output suppressed
!
end

telus1#

```

5.3.2 Results Analysis

We will now verify the implementation using the show commands and see how shaping queues are affected when video or voice data is passed through the traffic shaping channels. To view the shaping parameters configured inside the defined policy on the ISP router, use the “*show policy-map*” command in enable mode. The results are:

```
telus1#sh policy-map
```

```

Policy Map CBWFQ
Class voice
  Bandwidth 50 (%) Max Threshold 10 (packets)
  Traffic Shaping
    Peak Rate Traffic Shaping
      CIR 128000 (bps) Max. Buffers Limit 1000 (Packets)
Class data
  Bandwidth 40 (%) Max Threshold 20 (packets)
  Traffic Shaping
    Average Rate Traffic Shaping
      CIR 128000 (bps) Max. Buffers Limit 1000 (Packets)

```

Due to bandwidth limitation, we have to use a very low quality video for demonstration of this experiment. The video was transcoded down to 64 kbps. As discussed before, when video was played at the VLC server, a commendable quality was received at the client end. However, as soon as a call was injected in the channel (HeadOffice - BranchOffice), a humungous drop in video quality was noticed. However, the call quality was great.

Apparently, the queue for the “voice” class remains unoccupied and thus the shaping in that class remains inactive. However, the number of packets server by the queue kept increasing in number. Additionally, the queue in the “data” class started filling up immediately after the call was established with a noticeable amount of packet drop as well. This is because, the delay sensitive voice traffic was committed 50% fixed bandwidth that caused bandwidth starvation for the other data.

These results were captured using the “*show policy-map interface <interface>*” command recursively after regular intervals.

Telus1# show policy-map interface serial 0/0

Serial0/0

Service-policy output: CBWFQ

Class-map: **voice** (match-any)

Target/Average Rate	Byte Limit	Sustain bits/int	Excess bits/int	Interval (ms)	Increment (bytes)
256000/128000	1984	7936	7936	62	1984

Adapt	Queue	Packets	Bytes	Packets	Bytes	Shaping
Active	Depth			Delayed	Delayed	Active
-	0	26743	1711552	0	0	no

Class-map: **data** (match-all)

Target/Average Rate	Byte Limit	Sustain bits/int	Excess bits/int	Interval (ms)	Increment (bytes)
12800/12800	2000	8000	8000	666	1000

Adapt	Queue	Packets	Bytes	Packets	Bytes	Shaping
Active	Depth			Delayed	Delayed	Active
-	78	2584	3345577	47	51688	yes

Class-map: class-default (match-any)

2514 packets, 3153184 bytes
5 minute offered rate 0 bps, drop rate 0 bps
Match: any

Serial0/0

Service-policy output: CBWFQ

Class-map: **voice** (match-any)

Target/Average Rate	Byte Limit	Sustain bits/int	Excess bits/int	Interval (ms)	Increment (bytes)
256000/128000	1984	7936	7936	62	1984

Adapt	Queue	Packets	Bytes	Packets	Bytes	Shaping
Active	Depth			Delayed	Delayed	Active
-	0	27738	1775232	0	0	no

Class-map: **data** (match-all)

Target/Average Rate	Byte Limit	Sustain bits/int	Excess bits/int	Interval (ms)	Increment (bytes)
12800/12800	2000	8000	8000	666	1000

Adapt	Queue	Packets	Bytes	Packets	Bytes	Shaping
Active	Depth			Delayed	Delayed	Active
-	91	2611	3376333	74	82444	yes

Class-map: class-default (match-any)

2517 packets, 3153273 bytes
5 minute offered rate 0 bps, drop rate 0 bps
Match: any

6.3.3 Conclusion

In this experiment, we elaborated the concepts of Class Based Traffic Shaping and we demonstrated the implementation using our network model. Using CBTS, we can configure traffic shaping on a per-traffic-class basis, rather than doing it on the flow level. This allows us to classify the data in various classes and then allocate a specific amount of bandwidth to each of these classes. We have successfully established different priority levels for voice and other data using a different class for each of them.

Also, CBTS allows us to configure traffic shaping in a hierarchical policy map structure. We can apply traffic shaping on the parent policy and then define parameters on the various subclasses under them. As shown in figure 6.2, we also define a bunch of subclasses under the parent classes “voice” and “data”, depending upon the requirement. We can define up to 64 discrete classes using CBWFQ.

The queuing system used in this demonstration is Class Based Queuing. The default queuing system with CBTS is the WFQ. The “*bandwidth percent <percent>*” command changes it over to CBWFQ. The major drawback is that adaptive shaping is not supported in this mechanism.

6.4 Chapter 7 - Preview

The next chapter deals with another efficient mechanism for traffic shaping called Frame Relay Traffic Shaping. FRTS builds upon this existing Frame Relay support with additional capabilities that improve the scalability and performance of a Frame Relay network. The chapter starts with a brief outline of the Headlining features of frame relay networks followed by a discussion on FRTS concepts and implementation procedures. We then move on to the practical implementation of the technology on our Lab network model; analyzing the results and the behavior of the shaping queues.

.....

CHAPTER 7

Frame Relay Traffic Shaping

7.1 The Concept

Frame Relay is a Layer 2 (data link) wide-area network (WAN) protocol that works at both Layer 1 (physical) and Layer 2 (data link) of the OSI model. Frame Relay networks are typically deployed as a cost-effective replacement for point-to-point private line, or leased line, services

It is a packet-switched technology, meaning that each network end user, or end node, will share backbone network resources, such as bandwidth. Connectivity between these end nodes is accomplished with the use of Frame Relay virtual circuits (VCs). Two types of Frame Relay VCs exist:

- Permanent virtual circuits (PVCs)—These are permanently established, requiring no call setup, and utilize DLCIs for endpoint addressing.
- Switched virtual circuits (SVCs)—These are established as needed, requiring call setup procedures and utilizing X.121 or E.164 addresses for endpoint addressing.

Congestion detection is a very important aspect of any network protocol to ensure reliable and efficient delivery across the network. In frame Relay, two types of congestion-notification mechanisms are implemented: FECN and BECN.

Forward explicit congestion notification (FECN):

The FECN bit is set by a Frame Relay network to inform the Frame Relay networking device receiving the frame that congestion was experienced in the path from origination to destination. Frame relay network devices that receive frames with the FECN bit will act as directed by the upper-layer protocols in operation. The upper-layer protocols will initiate flow-control operations, depending on which upper-layer protocols are implemented. This flow-control action is typically the throttling back of data transmission, although some implementations can be designated to ignore the FECN bit and take no action.

Backward explicit congestion notification (BECN):

Much like the FECN bit, the BECN bit is set by a Frame Relay network to inform the DTE that is receiving the frame that congestion was experienced in the path traveling in the opposite direction of frames. The upper-layer protocols will initiate flow-control operations, depending on which upper-layer protocols are implemented. This flow-control action is

typically the throttling back of data transmission, although some implementations can be designated to ignore the BECN bit and take no action.

Another interesting capability of Frame Relay technology is that the frames can be specified regarding which have low priority or low time sensitivity. These frames will be the first to be dropped when a Frame Relay switch is congested and the mechanism that allows a Frame Relay switch to identify such frames is called the DISCARD ELIGIBLE BIT or the DE bit. DE lists and groups can be managed in the following manner:

- DE lists can be specified that identify the characteristics of frames to be eligible for discarding.
- DE groups can be specified to identify the affected DLCI.
- DE lists can also be specified based on the protocol or the interface, and on characteristics such as fragmentation of the packet, a specific TCP or User Datagram Protocol (UDP) port, an access list number, or a packet size.

Frame Relay Traffic Shaping:

FRTS builds upon this existing Frame Relay support with additional capabilities that improve the scalability and performance of a Frame Relay network, increasing the density of VCs and improving response time.

The use of traffic shaping is especially important in Frame Relay networks because the switch cannot determine which frames take precedence and therefore which frames should be dropped when congestion occurs. It is important for real-time traffic, such as VoFR, that latency be bounded, thereby bounding the amount of traffic and traffic loss in the data link network at any given time by keeping the data in the router that is making the guarantees. Retaining the data in the router allows the router to prioritize traffic according to the guarantees that the router is making. The processes involved in the operation of FRTS are pictorially represented in the following figure.

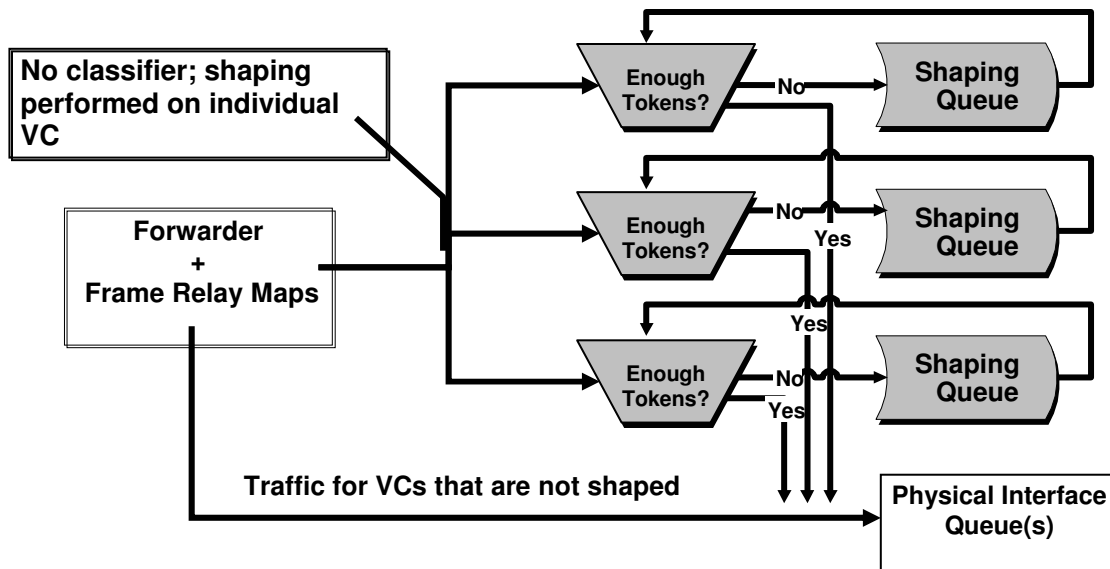


Figure 7.1

As is also true of GTS, FRTS can eliminate bottlenecks in Frame Relay networks that have high-speed connections at the central site and low-speed connections at branch sites. We can configure rate enforcement (a peak rate configured to limit outbound traffic) to limit the rate at which data is sent on the VC at the central site. This is exactly what we will try to accomplish in one of our experiments later in this section.

Frame Relay traffic shaping enables us to start a token bucket filter per Virtual Circuit in contrast to GTS in which we can only specify one peak rate (upper limit) per physical interface and one CIR (lower limit) value per sub-interface. Traffic shaping in frame relay limits the rate of transmission of data, limiting the data transfer to one of the following:

- A specific configured rate
- A *derived rate* based on the level of congestion

When an interface that is configured with traffic shaping receives a BECN, it immediately decreases, or throttles down, its maximum rate by a significant amount. If, after several intervals, the [throttled] interface has not received another BECN and traffic is waiting in the queue, the maximum rate slightly increases. This dynamically adjusted maximum rate is called the *derived rate*, which will always be between the upper bound and the lower bound that is configured on the interface.

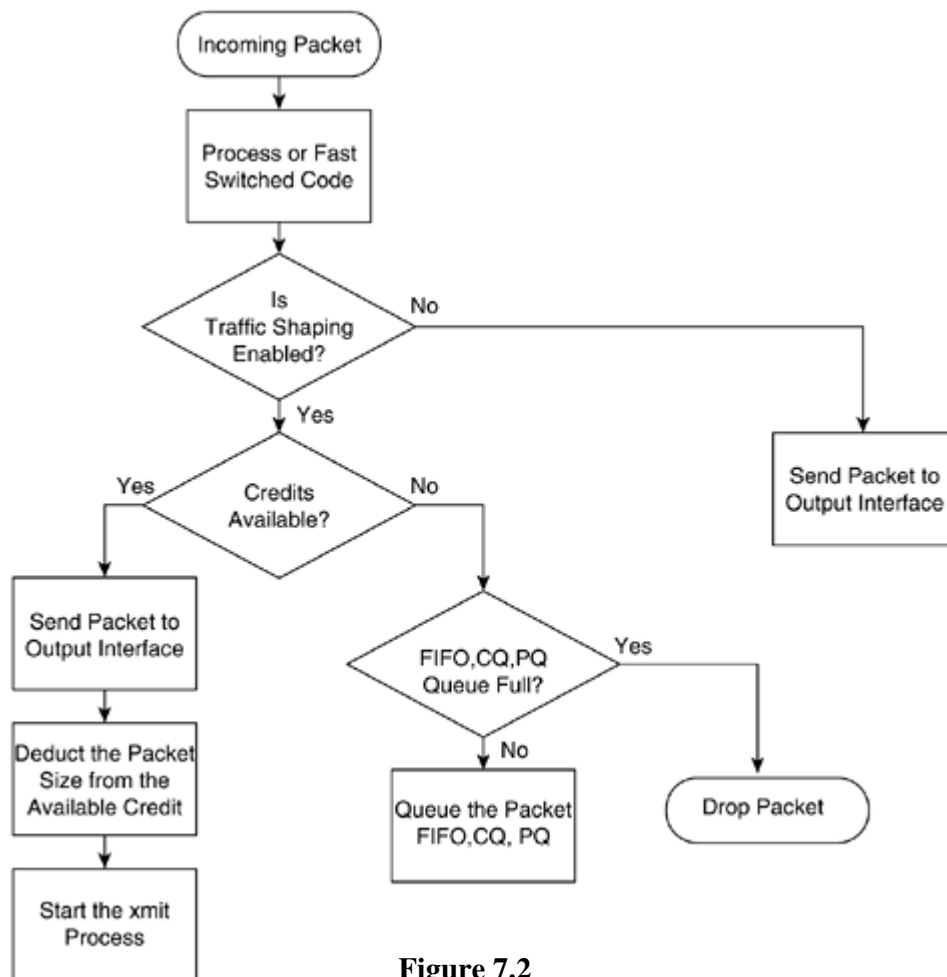


Figure 7.2

The ground process of Frame Relay traffic shaping is more or less same as the other shaping techniques with the difference of the queuing mechanisms used. Figure 7.2 gives a detailed vision of the steps followed by a frame relay shaper. In other words, this flow diagram details the flow of possibilities that could happen to a frame arriving at the interface of a router that has FRTS enabled.

7.1.1 Queuing

The queuing mechanisms supported with Frame Relay Traffic Shaping are Weighted Fair Queuing (WFQ), Priority Queuing (PQ), Custom Queuing (CQ), First-In-First-Out (FIFO). All these queues are implemented per VC. In our case, we will be using Priority Queuing with FRTS.

Priority queuing (PQ) enables network administrators to prioritize traffic based on specific criteria. These criteria include protocol or sub-protocol types, source interface, packet size, fragments, or any parameter through a standard or extended access list. PQ offers four different queues:

- Low priority
- Normal priority
- Medium priority
- High priority

Each packet is assigned to one of these queues. If no classification is assigned to a packet, it is placed in the normal priority queue.

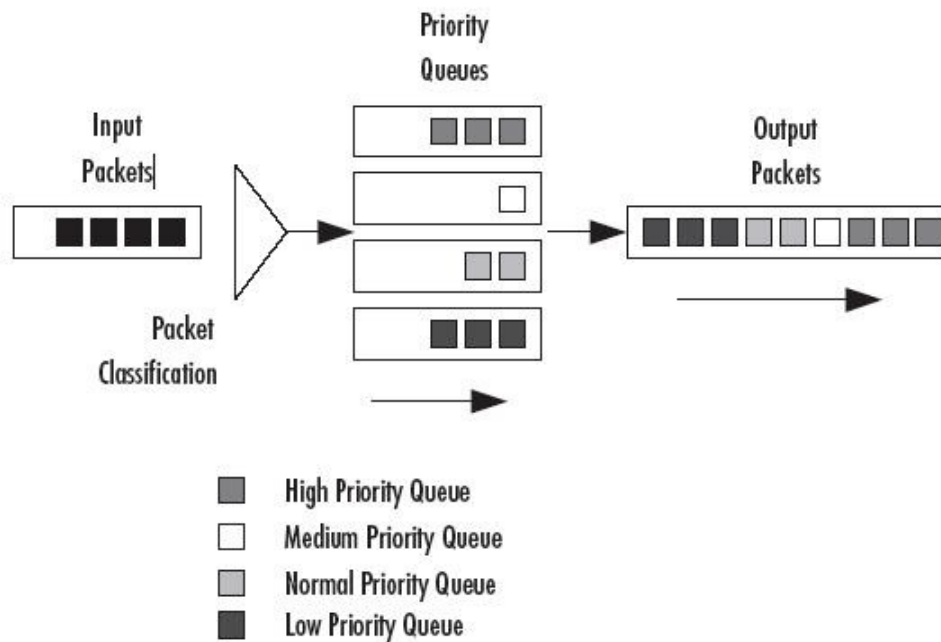


Figure 7.3

The priority of each queue is absolute. As packets are processed, PQ examines the state of each queue, always servicing higher priority queues before lower priority queues. This means that as long as there is traffic in a higher priority queue, lower priority queues will not be processed. PQ, therefore, does not use a fair allocation of resources among its queues. It services them strictly on the basis of the priority classifications configured by the network administrator. *Figure 7.2* shows PQ in action.

This can be noted that for delay sensitive real-time traffic like Voice, we can also use IP RTP priority to it strictly police the PQ contents when there is congestion on the interface. IP RTP Priority creates a strict priority queue on a Frame Relay PVC for a set of RTP packet flows that belong to a range of User Datagram Protocol (UDP) destination ports. This does not become active until there is congestion in the interface.

7.1.2 FRTS Features

To briefly nest the whole discussion above, we can say that traffic shaping over Frame Relay feature provides the following capabilities:

- Rate enforcement on a per-VC basis: You can configure a peak rate to limit outbound traffic to either the CIR or some other defined value such as the excess information rate (EIR).
- Generalized BECN support on a per-VC basis: The router can monitor BECNs and dynamically throttle traffic based on BECN-marked packet feedback from the Frame Relay network.
- Priority queuing (PQ), custom queuing (CQ) or WFQ support at the VC level. This allows for finer granularity in the prioritisation and queuing of traffic, giving you more control over the traffic flow on an individual VC. The traffic shaping over Frame Relay feature applies to Frame Relay permanent virtual circuits (PVCs) and switched virtual circuits (SVCs).
- Enhanced Local Management Interface (ELMI) enables automated exchange of Frame Relay QoS parameter information between the Cisco router and the Cisco switch. Routers can base congestion management and prioritization decisions on known QoS values, such as the CIR, Bc, and Be. The router senses QoS values from the switch and can be configured to use those values in traffic shaping. This enhancement works between Cisco routers and Cisco switches

7.1.3 Restrictions

The only prominent restriction that we can think of is that Frame Relay Traffic Shaping applies only to Frame Relay PVCs and SVCs.

7.2 Configuring FRTS

To configure Frame Relay Traffic Shaping, we need to define a map-class first. The map-class contains all the shaping and queuing parameters. Once the map-class is configured, Frame Relay Traffic Shaping can be started on any of the interfaces, sub-interfaces or virtual circuits. Listed below, are all the possible steps (optional or mandatory) that can constitute a legitimate FRTS implementation”

Step 1) Enable Frame Relay Encapsulation on an interface

```
interface interface-type interface-number  
encapsulation frame-relay [cisco | ietf]
```

Step 2) Enable Frame Relay Traffic Shaping on the Interface

```
interface interface-type interface-number  
frame-relay traffic-shaping
```

Step 3) Enable Enhanced Local Management Interface

```
interface interface-type interface-number  
frame-relay qos-autosense
```

Step 4) Specify a Traffic Shaping Map Class for the Interface

```
frame-relay class map-class-name
```

Step 5) Define a Map Class with Queueing and Traffic Shaping Parameters

```
map-class frame-relay map-class-name // Specify a map class to  
define.  
frame-relay traffic-rate average [peak] //Define the traffic rate for the  
map class.  
frame-relay custom-queue-list list-number //Specify a custom queue list.  
frame-relay priority-group list-number //Specify a priority queue list.  
frame-relay adaptive-shaping {becn | foresight} //Select either BECN or  
ForeSight as the congestion  
backward notification  
mechanism to which traffic  
shaping will adapt
```

Step 6) Define Access Lists

Step 7) Define Priority Queue Lists for the Map Class

```
priority-list list-number protocol protocol-name high | low | medium  
frame relay priority-group list-number
```

Step 8) Define Custom Queue Lists for the Map Class

```
queue-list list-number protocol protocol-name list access-list-number  
frame relay custom-queue-list list-number
```

Step 9) Create a Broadcast Queue for an Interface

```
frame-relay broadcast-queue size byte-rate packet-rate
//Create a broadcast queue for
an interface.
```

Step 10) Configure Discard Eligibility

```
frame-relay de-list list-number {protocol protocol | interface interface-type interface-
number} characteristic // Define a DE list
frame-relay de-group group-number dlci //Define a DE group.
```

Step 11) Configure DLCI Priority Levels

```
frame-relay priority-dlci-group group-number high-dlci medium-dlci normal-dlci
low-dlci
//Enable multiple parallel DLCIs for different
types of Frame Relay traffic, associate
specified DLCIs with the same group, and
define their levels
```

7.3 Simulation Study

In the simulation study, we are mainly interested in studying the change in behavior of the network after the implementation of the Frame Relay Traffic Shaping. We will consider our previously discussed LAB model (Figure 3.1), and try to implement the shaping mechanism in the Frame relay cloud. The queuing mechanism used will be Priority Queuing.

7.3.1 Experimental Setup

This scenario is approximately the same as the one discussed in the previous chapter with minute modifications. MINT Inc. decides to keep MINT-EDM1 as voice-only subnet and use the data and video on MINT-EDM2 subnet. Telus has an agreement with MINT Inc. to support voice and video streaming services through the frame relay network to the branch location at Calgary. We have a bandwidth limitation in the scenario that needs to be managed to support the above mentioned data types. The main concern is to allocate highest priority to the queue supporting the voice data over any other data. We are using the G.729 codec which would need, practically, 30 to 40 Kbps of channel capacity to be transmitted. On the other hand, video bandwidth depends on the quality of the video being transmitted.

Consider the network setup depicted in Figure 6.1. We will create a separate class for the voice traffic and use priority queuing to put it in the highest priority queue. Using classes we will also classify the other data in a separate class and allocate a throttled bandwidth to it as well.

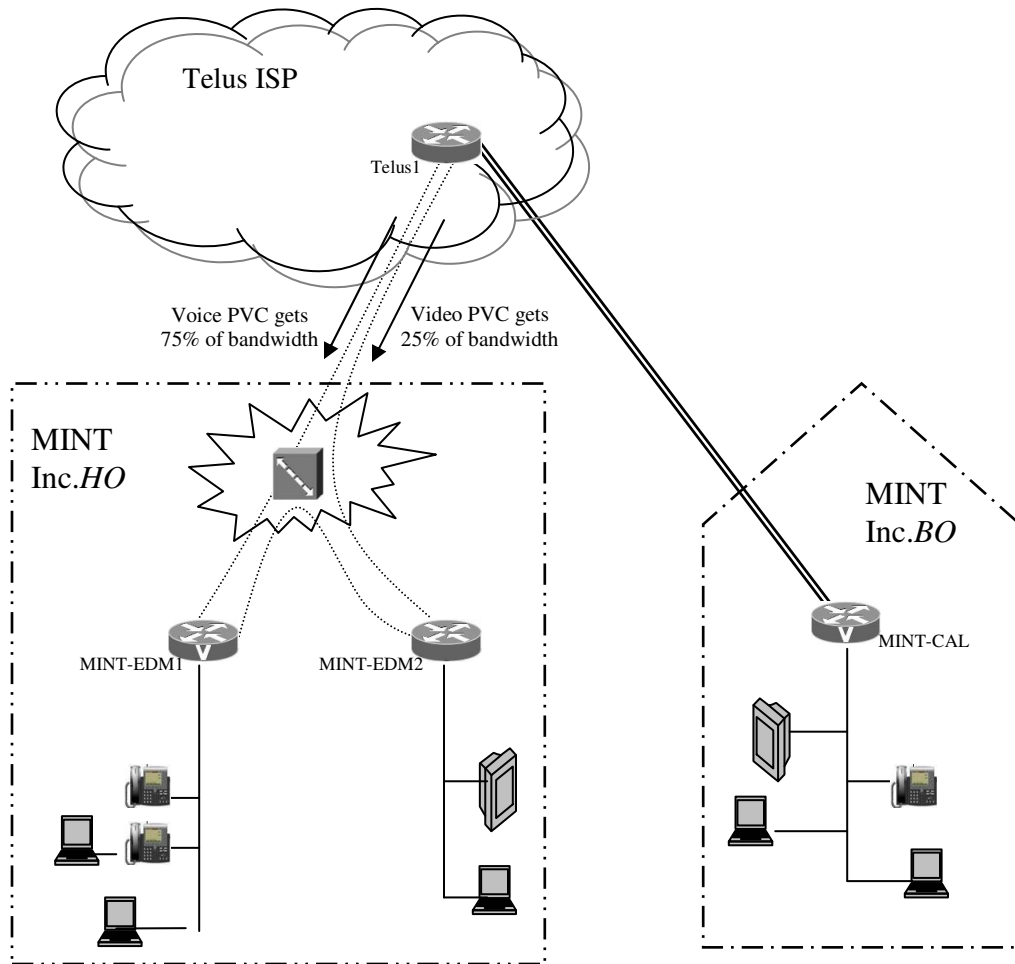


Figure 7.4

To implement this scenario, we defined two classes named “class 1” and “class 2” in a policy called “policy 1”. The DLCI 20 is associated with class 1, that will hold the voice data and the DLCI 22 is associated with class 2 that will hold the other data. These classes are also defined on the client routers MINT-EDM1 and MINT-EDM2. The configuration files for each of these devices are listed below:

Telus1:

```
telus1#
telus1#sh runn
Building configuration...

Current configuration : 2624 bytes
```

```

!
!
//output suppressed
!
class-map match-all class2
  match access-group 102
class-map match-all class1
  match access-group 101
!
!
policy-map policy1
  class class1
    bandwidth percent 75
  class class2
    bandwidth percent 25
!
!
!
interface Serial0/0.1 point-to-point
  bandwidth 10000000
  ip address 100.1.20.1 255.255.255.252
  frame-relay interface-dlci 22
  frame-relay class class2

!
interface Serial0/0.2 point-to-point
  ip address 100.1.10.1 255.255.255.252
  frame-relay interface-dlci 20
  frame-relay class class1
!
map-class frame-relay class2
  frame-relay cir 64000
  frame-relay bc 8000
  frame-relay be 1000
  frame-relay mincir 64000
  service-policy output policy1
!
map-class frame-relay class1
  frame-relay cir 64000
  frame-relay bc 8000
  frame-relay mincir 32000
  frame-relay adaptive-shaping becn
  service-policy output policy1
  frame-relay priority-group 2
!
access-list 101 permit udp any any range 16384 32767
access-list 102 permit tcp any any
priority-list 2 protocol udp high list 101
!
!
//output suppressed
!
end

telus1#

```


MINT-EDM1:

```
MINT-EDM1#
MINT-EDM1#sh run
Building configuration...
!
!
//output suppressed
!
interface Serial0/1.1 point-to-point
 ip address 100.2.10.2 255.255.255.252
 frame-relay interface-dlci 17
!
interface Serial0/1.2 point-to-point
 ip address 100.1.10.2 255.255.255.252
 frame-relay class class1
 frame-relay interface-dlci 21
!
!
!
map-class frame-relay class1
 frame-relay cir 64000
 frame-relay bc 8000
 frame-relay mincir 32000
 frame-relay adaptive-shaping becn
!
//output suppressed
!
end

MINT-EDM1#
```

MINT-EDM2:

```
MINT-EDM2#sh run
Building configuration...

Current configuration : 1769 bytes
!
!
//output suppressed
!
interface Serial0/1.1 point-to-point
 bandwidth 10000000
 ip address 100.1.20.2 255.255.255.252
 frame-relay class class2
 frame-relay interface-dlci 23
!
!
!
map-class frame-relay class2
 frame-relay cir 64000
 frame-relay bc 8000
 frame0relay be 1000
```

```

frame-relay mincir 64000
frame-relay adaptive-shaping becn
!
//output suppressed
!
end

```

MINT-EDM2#

7.3.2 Results Analysis

Let us verify the implementation using the show commands and see how shaping queues are affected when video or voice data is passed through the traffic shaping channels. To view the shaping parameters configured inside the defined policy on the ISP router, use the “*show policy-map interface <intf-type>*” command in enable mode:

```
telus1#sh policy-map interface serial 0/0.2
```

```
Serial0/0.2: DLCI 20 -
```

```
Service-policy output: policy1
```

```

Class-map: class1 (match-all)
  27089 packets, 1744260 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
Match: access-group 101
Queueing
  Output Queue: Conversation 25
  Bandwidth 75 (%) Max Threshold 64 (packets)
  (pkts matched/bytes matched) 0/0
  (depth/total drops/no-buffer drops) 0/0/0

```

```

Class-map: class-default (match-any)
  18551 packets, 15645233 bytes
  5 minute offered rate 117000 bps, drop rate 53000 bps
Match: any

```

```
Serial0/0.2: DLCI 20 -
```

```
Service-policy output: policy1
```

```

Class-map: class1 (match-all)
  28786 packets, 1853749 bytes
  5 minute offered rate 8000 bps, drop rate 0 bps
Match: access-group 101
Queueing
  Output Queue: Conversation 25
  Bandwidth 75 (%) Max Threshold 64 (packets)
  (pkts matched/bytes matched) 0/0

```

```
(depth/total drops/no-buffer drops) 0/0/0
```

```
Class-map: class-default (match-any)  
  28432 packets, 18184951 bytes  
  5 minute offered rate 109000 bps, drop rate 55000 bps  
Match: any
```

Video and voice data were transmitted concurrently over the network. The voice data will be transmitted over the virtual circuit with DLCI combination 20-21 and the video data will be transmitted over the VC with DLCI combination 22-23. The voice class has enough bandwidth to support the voice data using the G.729 codec and hence will experience no packet drops. However, the unclassified packets in the priority queuing that fall under the default queue, will experience a substantial amount of drop rate due bandwidth starving. This can be verified on the VC-level by using the “*show frame-relay pvc interface <intf-type> <dcli-number>*” command:

```
telus1#show frame-relay pvc interface serial 0/0.2 20
```

```
PVC Statistics for interface Serial0/0.2 (Frame Relay DTE)  
  
DLCI = 20, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =  
Serial0/0.2  
  
input pkts 65796           output pkts 56190           in bytes 68277558  
out bytes 18249713         dropped pkts 0               in pkts dropped 0  
out pkts dropped 9533      out bytes dropped 12922082  
late-dropped out pkts 9533    late-dropped out bytes 12922082  
in FECN pkts 0             in BECN pkts 0               out FECN pkts 0  
out BECN pkts 0            in DE pkts 0                 out DE pkts 0  
out bcast pkts 208         out bcast bytes 50076  
Shaping adapts to BECN  
pvc create time 02:48:13, last time pvc status changed 02:28:23  
cir 64000      bc 8000      be 0          byte limit 1000  interval 125  
mincir 32000   byte increment 1000 Adaptive Shaping BECN  
pkts 55724     bytes 18132824  pkts delayed 21966   bytes delayed  
15843113  
shaping active  
traffic shaping drops 9533  
service policy policy1  
Serial0/0.2: DLCI 20 -  
  
Service-policy output: policy1  
  
Class-map: class1 (match-all)  
37008 packets, 2383233 bytes  
  5 minute offered rate 0 bps, drop rate 0 bps  
Match: access-group 101  
Queueing  
  Output Queue: Conversation 25  
  Bandwidth 75 (%) Max Threshold 64 (packets)  
  (pkts matched/bytes matched) 0/0  
  (depth/total drops/no-buffer drops) 0/0/0
```

```

Class-map: class-default (match-any)
  28313 packets, 28773489 bytes
  5 minute offered rate 117000 bps, drop rate 53000 bps
  Match: any
  Output queue size 64/max total 600/drops 9581

```

```

PVC Statistics for interface Serial0/0.2 (Frame Relay DTE)

DLCI = 20, DLCI USAGE = LOCAL, PVC STATUS = ACTIVE, INTERFACE =
Serial0/0.2

  input pkts 65799           output pkts 56327           in bytes 68277709
  out bytes 18433420         dropped pkts 0               in pkts dropped 0
  out pkts dropped 9646      out bytes dropped 13075762
  late-dropped out pkts 9646  late-dropped out bytes 13075762
  in FECN pkts 0            in BECN pkts 0              out FECN pkts 0
  out BECN pkts 0           in DE pkts 0                out DE pkts 0
  out bcst pkts 208         out bcst bytes 50076
  Shaping adapts to BECN
  pvc create time 02:48:36, last time pvc status changed 02:28:46
  cir 64000      bc 8000      be 0      byte limit 1000  interval 125
  mincir 32000   byte increment 1000 Adaptive Shaping BECN
  pkts 55862     bytes 18317891  pkts delayed 22104   bytes delayed
16028180
  shaping active
  traffic shaping drops 9646
  service policy policy1
Serial0/0.2: DLCI 20 -

Service-policy output: policy1

Class-map: class1 (match-all)
  37008 packets, 2383233 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: access-group 101
  Queueing
    Output Queue: Conversation 25
    Bandwidth 75 (%) Max Threshold 64 (packets)
    (pkts matched/bytes matched) 0/0
    (depth/total drops/no-buffer drops) 0/0/0

Class-map: class-default (match-any)
  28564 packets, 29112236 bytes
  5 minute offered rate 117000 bps, drop rate 53000 bps
  Match: any
  Output queue size 64/max total 600/drops 9658
telus1#

```

It is clearly evident that the number of packets in both the classes keeps increasing. Apparently, the voice class “class1” doesn’t notice and packet loss whereas the unclassified data had huge drop rate. Voice quality was good. Thus, voice was given priority over the rest of the data.

7.3.3 Conclusion

The most significant advantage of Frame Relay Traffic Shaping is that it enables us to start a token bucket filter per Virtual Circuit rather than getting restricted to the interface level shaping. This makes it the most appropriate selection to go with Frame Relay networks. The main utility of FRTS is exploited in Multilink (Point-to-Multipoint) frame-relay networks.

We have successfully implemented priority queuing in the above experiment to allocate the highest priority to the voice traffic making sure that there is no delay or jitter that might affect the voice quality. The following statistics from the VoIP phone reveal the expected results:

MINT-EDM1# debug ephone statistics mac-address 0007.8505.631D

```
EPHONE statistics debugging is enabled for phone 0007.8505.631D
MINT-EDM1#
*Mar 1 00:37:28.079: ephone-1[1]:Call Stats for line 3 DN 1 1001 ref 3
*Mar 1 00:37:28.079: ephone-1[1]:TX Pkts 7824 bytes 250091 RX Pkts
1595 bytes 43999
*Mar 1 00:37:28.079: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
*Mar 1 00:37:28.079: ephone-1[1]:Src 100.4.0.6 27928 Dst 100.4.0.1
2000 bytes 20 vad 250 G729
*Mar 1 00:37:30.763: STATS: DN 1 Packets Sent 7824
*Mar 1 00:37:30.763: STATS: DN 1 Packets Received 1595
*Mar 1 00:37:30.763: STATS: DN 1 Latency 0
*Mar 1 00:37:30.763: STATS: DN 1 PacketsLost 0
*Mar 1 00:37:32.619: ephone-1[1]:GetCallStats line 3 ref 3 DN 1: 1001
*Mar 1 00:37:32.871: ephone-1[1]:Call Stats for line 3 DN 1 1001 ref 3
*Mar 1 00:37:32.871: ephone-1[1]:TX Pkts 8064 bytes 257771 RX Pkts
1608 bytes 44168
*Mar 1 00:37:32.871: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
*Mar 1 00:37:32.871: ephone-1[1]:Src 100.4.0.6 27928 Dst 100.4.0.1
2000 bytes 20 vad 250 G729
*Mar 1 00:37:34.819: STATS: DN 1 Packets Sent 8064
*Mar 1 00:37:34.819: STATS: DN 1 Packets Received 1608
*Mar 1 00:37:34.819: STATS: DN 1 Latency 0
*Mar 1 00:37:34.819: STATS: DN 1 PacketsLost 0
*Mar 1 00:37:37.663: ephone-1[1]:GetCallStats line 3 ref 3 DN 1: 1001
*Mar 1 00:37:37.915: ephone-1[1]:Call Stats for line 3 DN 1 1001 ref 3
*Mar 1 00:37:37.915: ephone-1[1]:TX Pkts 8316 bytes 265835 RX Pkts
1619 bytes 44311
*Mar 1 00:37:37.915: ephone-1[1]:Pkts lost 0 jitter 0 latency 0
*Mar 1 00:37:37.915: ephone-1[1]:Src 100.4.0.6 27928 Dst 100.4.0.1
2000 bytes 20 vad 250 G729
*Mar 1 00:37:41.443: STATS: DN 1 Packets Sent 8316
*Mar 1 00:37:41.443: STATS: DN 1 Packets Received 1619
*Mar 1 00:37:41.443: STATS: DN 1 Latency 0
*Mar 1 00:37:41.443: STATS: DN 1 PacketsLost 0
```

.....

APPENDIX - I

Configuration Files

This section includes the Configuration Files of all the networking components that our Network Model is comprised of. Please be informed that these configurations implement the scenario shown in figure 3.1 of this report and does not include the traffic shaping tests that have been performed on them during the life cycle of the project. The configuration commands used to implement the Traffic shaping techniques are detailed in the respective discussions about each. Please refer to the appropriate chapter for these additional commands.

The following devices' configuration has been archived using the “*show running configuration*” command at the *enable* mode on each box:

- {A} ‘Telus1’ router – The Primary router for the primary ISP, TELUS
(Cisco 2800 series Integrated Services router)
- {B} ‘Telus2’ router – The Backup router for the primary ISP, TELUS
(Cisco 3600 Series multi-service access platform router)
- {C} ‘Shaw’ router – The only router in the backup ISP, SHAW
(Cisco 2800 series Integrated Services router)
- {D} ‘FR-SW’ frame-relay switch – Contrives the Frame Relay cloud for TELUS
(Cisco 3600 Series multi-service access platform router)
- {E} ‘MINT-EDM1’ router – Voice enabled router for Head office, hosting Voice+Data
(Cisco 2600 series Multi-service Platform router)
- {F} ‘MINT-EDM2’ router – Second router in the Head Office, hosting Video+Data
(Cisco 2600 series Multi-service Platform router)
- {G} ‘MINT-CAL’ router – The router at the Branch Office, hosting Voice+Video+Data
(Cisco 2600 series Multi-service Platform router)
- {H} ‘Switch’ switch – Splits the Local network at Head office into two VLANs
(Catalyst 3500 Series)

{A} 'Telus1' router

```
telus1#sh run
Building configuration...

Current configuration : 2129 bytes
!
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname telus1
!
!
ip subnet-zero
!
!
!
ip cef
call rsvp-sync
!
!
!
interface Serial0/0
 bandwidth 10000000
 no ip address
 encapsulation frame-relay
 no fair-queue
 clock rate 128000
!
interface Serial0/0.1 point-to-point
 bandwidth 10000000
 ip address 100.1.20.1 255.255.255.252
 frame-relay interface-dlci 22
!
interface Serial0/0.2 point-to-point
 ip address 100.1.10.1 255.255.255.252
 frame-relay interface-dlci 20
!
interface Serial0/1
 bandwidth 10000000
 ip address 100.5.0.1 255.255.255.252
!
interface Serial0/2
 bandwidth 10000000
 ip address 150.0.0.1 255.255.255.252
 clock rate 128000
!
interface Serial0/3
 no ip address
 shutdown
 clock rate 64000
!
interface Serial0/4
 no ip address
```

```

    shutdown
!
interface Serial0/5
    no ip address
    shutdown
!
interface Serial0/6
    no ip address
    shutdown
!
interface Serial0/7
    no ip address
    shutdown
!
interface Ethernet1/0
    ip address 100.0.0.1 255.255.255.252
    half-duplex
!
interface ATM2/0
    no ip address
    shutdown
    no atm ilmi-keepalive
!
router bgp 100
    bgp log-neighbor-changes
    network 100.0.0.0 mask 255.255.255.252
    network 100.1.10.0 mask 255.255.255.252
    network 100.1.20.0 mask 255.255.255.252
    network 100.5.0.0 mask 255.255.255.252
    network 150.0.0.0 mask 255.255.255.252
    aggregate-address 100.0.0.0 255.0.0.0 summary-only
    neighbor 100.0.0.2 remote-as 100
    neighbor 100.1.10.2 remote-as 1001
    neighbor 100.1.10.2 description Head-Office Voice Gateway (Primary
Link)
    neighbor 100.1.10.2 default-originate
    neighbor 100.1.10.2 route-map med4primary out
    neighbor 100.1.20.2 remote-as 1001
    neighbor 100.1.20.2 description Head-Office Server Gateway (Primary
Link)
    neighbor 100.1.20.2 default-originate
    neighbor 100.1.20.2 route-map med4primary out
    neighbor 100.5.0.2 remote-as 1002
    neighbor 100.5.0.2 default-originate
    neighbor 150.0.0.2 remote-as 200
!
ip classless
ip http server
!
route-map med4primary permit 10
    set metric 100
!
!
!
dial-peer cor custom
!
!

```



```
!  
gateway  
!  
!  
line con 0  
line aux 0  
line vty 0 4  
  login  
!  
end  
  
telus1#
```

{B} 'Telus2' router

```
telus2#sh run
Building configuration...

Current configuration : 1601 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname telus2
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
!
resource policy
!
ip subnet-zero
!
!
ip cef
!
!
voice-card 0
  no dspfarm
!
!
interface GigabitEthernet0/0
  ip address 100.0.0.2 255.255.255.252
  duplex auto
  speed auto
!
interface GigabitEthernet0/1
  no ip address
  shutdown
  duplex auto
  speed auto
!
interface Serial0/0/0
  bandwidth 10000000
  no ip address
  encapsulation frame-relay
  no fair-queue
!
interface Serial0/0/0.1 point-to-point
  ip address 100.2.10.1 255.255.255.252
  frame-relay interface-dlci 16
!
interface Serial0/0/0.2 point-to-point
  ip address 100.2.20.1 255.255.255.252
  frame-relay interface-dlci 18
```

```

!
router bgp 100
  no synchronization
  bgp log-neighbor-changes
  network 100.2.10.0 mask 255.255.255.252
  neighbor 100.0.0.1 remote-as 100
  neighbor 100.2.10.2 remote-as 1001
  neighbor 100.2.10.2 description Head-Office Voice Gateway
  neighbor 100.2.10.2 default-originate
  neighbor 100.2.10.2 route-map med4backup out
  neighbor 100.2.20.2 remote-as 1001
  neighbor 100.2.20.2 description Head-Office Video Server
  neighbor 100.2.20.2 default-originate
  neighbor 100.2.20.2 route-map med4backup out
  no auto-summary
!
ip classless
!
!
ip http server
no ip http secure-server
!
!
route-map med4backup permit 10
  set metric 200
!
!
control-plane
!
!
!
gateway
  timer receive-rtsp 1200
!
line con 0
line aux 0
line vty 0 4
  login
!
scheduler allocate 20000 1000
!
end

```

{C} 'Shaw' router

```
shaw#sh run
Building configuration...

Current configuration : 1055 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname shaw
!
boot-start-marker
boot-end-marker
!
!
no aaa new-model
!
resource policy
!
ip subnet-zero
!
!
ip cef
!
!
!
voice-card 0
  no dspfarm

!
interface GigabitEthernet0/0
  no ip address
  shutdown
  duplex auto
  speed auto
!
interface GigabitEthernet0/1
  ip address 200.0.0.1 255.255.255.252
  duplex auto
  speed auto
!
interface Serial0/0/0
  ip address 150.0.0.2 255.255.255.252
!
router bgp 200
  no synchronization
  bgp log-neighbor-changes
  network 150.0.0.0 mask 255.255.255.252
  network 200.0.0.0 mask 255.255.255.252
  aggregate-address 200.0.0.0 255.255.0.0 summary-only
  neighbor 150.0.0.1 remote-as 100
  neighbor 200.0.0.2 remote-as 1002
  neighbor 200.0.0.2 default-originate
```

```
no auto-summary
!  
ip classless
!  
!  
ip http server  
no ip http secure-server  
!  
!  
control-plane  
!  
!  
line con 0  
line aux 0  
line vty 0 4  
  login  
!  
scheduler allocate 20000 1000  
!  
end  
  
shaw#
```

{D} 'FR-SW' Frame relay switch

```
FR-SW#sh run
Building configuration...

Current configuration : 1579 bytes
!
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname FR-SW
!
!
ip subnet-zero
!
!
!
frame-relay switching
call rsvp-sync
!
!
!
interface Ethernet0/0
  no ip address
  shutdown
  half-duplex
!
interface Serial1/0
  bandwidth 10000000
  no ip address
  encapsulation frame-relay
  no fair-queue
  frame-relay intf-type dce
  frame-relay route 20 interface Serial1/2 21
  frame-relay route 22 interface Serial1/3 23
!
interface Serial1/1
  bandwidth 10000000
  no ip address
  encapsulation frame-relay
  clock rate 128000
  frame-relay intf-type dce
  frame-relay route 16 interface Serial1/2 17
  frame-relay route 18 interface Serial1/3 19
!
interface Serial1/2
  bandwidth 10000000
  no ip address
  encapsulation frame-relay
  clock rate 128000
  frame-relay intf-type dce
  frame-relay route 17 interface Serial1/1 16
  frame-relay route 21 interface Serial1/0 20
```

```

    frame-relay route 24 interface Serial1/3 25
    !
interface Serial1/3
    bandwidth 10000000
    no ip address
    encapsulation frame-relay
    frame-relay intf-type dce
    frame-relay route 19 interface Serial1/1 18
    frame-relay route 23 interface Serial1/0 22
    frame-relay route 25 interface Serial1/2 24
    !
interface Serial1/4
    no ip address
    shutdown
    !
interface Serial1/5
    no ip address
    shutdown
    !
interface Serial1/6
    no ip address
    shutdown
    !
interface Serial1/7
    no ip address
    shutdown
    !
interface ATM2/0
    no ip address
    shutdown
    no atm ilmi-keepalive
    !
ip classless
ip http server
    !
    !
    !
dial-peer cor custom
    !
    !
    !
line con 0
line aux 0
line vty 0 4
    login
    !
end

```

{E} 'MINT-EDM1' router

```
MINT-EDM1#sh runn
Building configuration...

Current configuration : 3153 bytes
!
version 12.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname MINT-EDM1
!
logging queue-limit 100
!
clock timezone mst -7
clock summer-time pst recurring
ip subnet-zero
!
!
no ip dhcp conflict logging
ip dhcp excluded-address 100.4.0.1 100.4.0.5
!
ip dhcp pool MINT
    network 100.4.0.0 255.255.0.0
    default-router 100.4.0.1
    option 150 ip 100.4.0.1
    domain-name Head-Office
!
mpls ldp logging neighbor-changes
!
!
no voice hpi capture buffer
no voice hpi capture destination
!
!
mta receive maximum-recipients 0
!
!
interface Loopback1
    ip address 100.10.0.1 255.255.255.0
!
interface FastEthernet0/0
    ip address 100.4.0.1 255.255.0.0
    duplex auto
    speed auto
!
interface Serial0/0
    no ip address
    shutdown
    no fair-queue
!
interface FastEthernet0/1
    no ip address
```



```

shutdown
duplex auto
speed auto
!
interface Serial0/1
bandwidth 10000000
no ip address
encapsulation frame-relay
!
interface Serial0/1.1 point-to-point
ip address 100.2.10.2 255.255.255.252
frame-relay interface-dlci 17
!
interface Serial0/1.2 point-to-point
ip address 100.1.10.2 255.255.255.252
frame-relay interface-dlci 21
!
interface Serial0/1.3 point-to-point
ip address 100.3.0.2 255.255.255.252
frame-relay interface-dlci 24
!
router bgp 1001
no synchronization
bgp log-neighbor-changes
network 100.2.10.0 mask 255.255.255.252
network 100.3.0.0 mask 255.255.255.252
network 100.4.0.0 mask 255.255.0.0
network 100.10.0.0 mask 255.255.255.0
neighbor 100.1.10.1 remote-as 100
neighbor 100.1.10.1 route-map lpref4primary out
neighbor 100.2.10.1 remote-as 100
neighbor 100.2.10.1 route-map lpref4backup out
neighbor 100.3.0.1 remote-as 1001
no auto-summary
!
ip http server
ip classless
!
!
!
!
route-map lpref4primary permit 10
set local-preference 200
!
route-map lpref4backup permit 10
set local-preference 90
!
!
tftp-server flash:P00303020209.bin
call rsvp-sync
!
voice-port 1/0/0
!
voice-port 1/0/1
!
!
mgcp profile default

```

```

!
dial-peer cor custom
!
!
!
dial-peer voice 1 pots
  destination-pattern 1002
  port 1/0/0
!
dial-peer voice 1000 voip
  preference 1
  destination-pattern 2001
  session target ipv4:200.0.1.1
!
dial-peer voice 2 pots
  destination-pattern 1012
  port 1/0/0
!
dial-peer voice 3 pots
  destination-pattern 1001
  port 1/0/1
!
dial-peer voice 4 pots
  destination-pattern 1011
  port 1/0/1
!
dial-peer voice 2000 voip
  preference 1
  destination-pattern 2011
  session target ipv4:200.0.1.1
!
gateway
!
!
telephony-service
  load 7960-7940 P00303020209
  max-ephones 4
  max-dn 4
  ip source-address 100.4.0.1 port 2000
  create cnf-files version-stamp Jan 01 2002 00:00:00
  keepalive 45
!
!
ephone-dn 1
  number 1001
  name Head Office 1
!
!
ephone-dn 2
  number 1011
  name neyota, kenny
!
!
ephone-dn 3
  number 1002
  name Head Office
!

```

```
!  
ephone-dn 4  
  number 1012  
  name gill, gurvir  
!  
!  
ephone 1  
  keepalive 45  
  mac-address 0007.8505.631D  
  type 7960  
  button 2:2 3:1  
!  
!  
!  
ephone 2  
  keepalive 45  
  mac-address 0007.50D5.4FA6  
  type 7960  
  button 1:3 2:4  
!  
!  
!  
line con 0  
line aux 0  
line vty 0 4  
  login  
!  
ntp server 100.4.0.1  
!  
end
```

MINT-EDM1#

{F} 'MINT-EDM2' router

```
MINT-EDM2#sh run
Building configuration...

Current configuration : 1769 bytes
!
version 12.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname MINT-EDM2
!
logging queue-limit 100
!
ip subnet-zero
!
!
!
mpls ldp logging neighbor-changes
!
!
no voice hpi capture buffer
no voice hpi capture destination
!
!
mta receive maximum-recipients 0
!
!
!
interface Loopback1
 ip address 100.7.0.1 255.255.0.0
!
interface FastEthernet0/0
 ip address 100.6.0.1 255.255.0.0
 duplex auto
 speed auto
!
interface Serial0/0
 no ip address
 shutdown
 no fair-queue
!
interface Serial0/1
 bandwidth 10000000
 no ip address
 encapsulation frame-relay
 clockrate 128000
!
interface Serial0/1.1 point-to-point
 bandwidth 10000000
 ip address 100.1.20.2 255.255.255.252
 frame-relay interface-dlci 23
!
interface Serial0/1.2 point-to-point
```

```

ip address 100.2.20.2 255.255.255.252
frame-relay interface-dlci 19
!
interface Serial0/1.3 point-to-point
ip address 100.3.0.1 255.255.255.252
frame-relay interface-dlci 25
!
router bgp 1001
no synchronization
bgp log-neighbor-changes
network 100.2.20.0 mask 255.255.255.252
network 100.3.0.0 mask 255.255.255.252
network 100.6.0.0 mask 255.255.0.0
network 100.7.0.0 mask 255.255.0.0
neighbor 100.1.20.1 remote-as 100
neighbor 100.1.20.1 route-map lpref4primary out
neighbor 100.2.20.1 remote-as 100
neighbor 100.2.20.1 route-map lpref4backup out
neighbor 100.3.0.2 remote-as 1001
no auto-summary
!
ip http server
ip classless
!
!
route-map lpref4primary permit 10
set local-preference 200
!
route-map lpref4backup permit 10
set local-preference 90
!
!
call rsvp-sync
!
voice-port 1/0/0
!
voice-port 1/0/1
!
!
mgcp profile default
!
dial-peer cor custom
!
!
line con 0
line aux 0
line vty 0 4
login
!
!
end

```

MINT-EDM2#

{G} 'MINT-EDM2' router

```
MINT-CAL#sh runn
Building configuration...

Current configuration : 3012 bytes
!
version 12.2
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname MINT-CAL
!
logging queue-limit 100
!
clock timezone mst -7
clock summer-time pst recurring
ip subnet-zero
!
!
no ip dhcp conflict logging
ip dhcp excluded-address 100.8.0.1 100.8.0.5
!
ip dhcp pool MINT2
    network 100.8.0.0 255.255.0.0
    default-router 100.8.0.1
    option 150 ip 100.8.0.1
    domain-name Branch-Office
!
mpls ldp logging neighbor-changes
!
!
!
no voice hpi capture buffer
no voice hpi capture destination
!
!
mta receive maximum-recipients 0
!
!
interface Loopback0
    ip address 4.4.4.4 255.255.255.255
!
interface Loopback1
    ip address 200.0.1.1 255.255.255.0
!
interface FastEthernet0/0
    ip address 100.8.0.1 255.255.0.0
    duplex auto
    speed auto
!
interface Serial0/0
    no ip address
    shutdown
```

```

    no fair-queue
    !
interface FastEthernet0/1
    ip address 200.0.0.2 255.255.255.252
    duplex auto
    speed auto
    !
interface Serial0/1
    bandwidth 10000000
    ip address 100.5.0.2 255.255.255.252
    clockrate 128000
    !
router ospf 1
    log-adjacency-changes
    network 4.4.4.4 0.0.0.0 area 2
    network 192.168.2.0 0.0.0.255 area 2
    !
router bgp 1002
    no synchronization
    bgp log-neighbor-changes
    network 100.5.0.0 mask 255.255.255.252
    network 100.8.0.0 mask 255.255.0.0
    network 200.0.0.0 mask 255.255.255.252
    network 200.0.1.0
    neighbor 100.5.0.1 remote-as 100
    neighbor 100.5.0.1 route-map lpref4telus out
    neighbor 200.0.0.1 remote-as 200
    neighbor 200.0.0.1 route-map lpref4shaw out
    no auto-summary
    !
ip http server
ip classless
!
!
!
!
route-map lpref4telus permit 10
    set local-preference 200
    !
route-map lpref4shaw permit 10
    set local-preference 90
    !
!
tftp-server flash:P00303020209.bin
call rsvp-sync
!
voice-port 1/0/0
!
voice-port 1/0/1
!
!
mgcp profile default
!
dial-peer cor custom
!
!
!

```

```

dial-peer voice 1 pots
 destination-pattern 2001
 port 1/0/0
!
dial-peer voice 2000 voip
 preference 1
 destination-pattern 1002
 session target ipv4:100.10.0.1
!
dial-peer voice 2 pots
 destination-pattern 2011
 port 1/0/0
!
dial-peer voice 3000 voip
 preference 2
 destination-pattern 1012
 session target ipv4:100.10.0.1
!
dial-peer voice 4000 voip
 preference 3
 destination-pattern 1001
 session target ipv4:100.10.0.1
!
dial-peer voice 5000 voip
 preference 4
 destination-pattern 1011
 session target ipv4:100.10.0.1
!
!
telephony-service
 load 7960-7940 P00303020209
 max-ephones 4
 max-dn 4
 ip source-address 100.8.0.1 port 2000
 create cnf-files version-stamp Jan 01 2002 00:00:00
 keepalive 45
!
!
ephone-dn 1
 number 2001
 name Branch Office
!
!
ephone-dn 2
 number 2011
 name neyota, kenny
!
!
ephone-dn 3
 number 2002
 name Branch Office
!
!
ephone-dn 4
 number 2012
 name gill, gurvir
!

```



```
!  
ephone 1  
  keepalive 45  
  mac-address 0007.5079.BF61  
  type 7960  
  button 1:1 2:2  
!  
!  
!  
ephone 2  
  keepalive 45  
  mac-address 0007.8513.13B8  
  type 7960  
  button 1:3 2:4  
!  
!  
!  
line con 0  
line aux 0  
line vty 0 4  
  login  
!  
ntp server 100.8.0.1  
!  
end
```

MINT-CAL#

APPENDIX - II

The following representation shows the various phases the project has evolved through to make its way to completion:

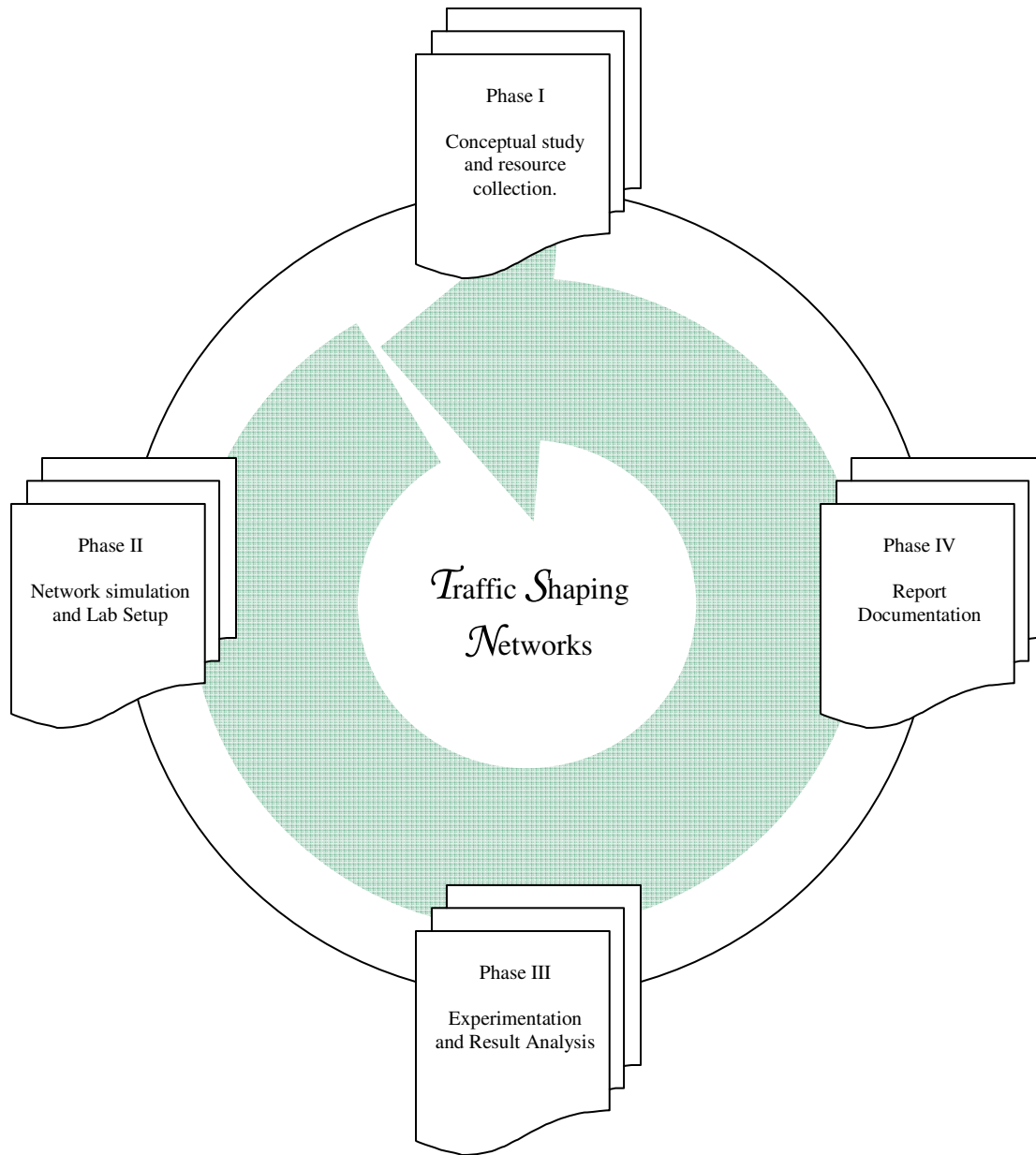


Figure A1

BIBLIOGRAPHY

- [1] Title: “Queuing and Congestion Avoidance Overview”
Author: Michael E. Flannagan, Benoit Durand, Jerry Sommerville, Mark Buchmann and Ron Fuller..
- [2] Title: CCNP: Building Scalable Cisco Internetworks Study Guide.
Author: Carl Timm and Wade Edwards.
- [3] Title: CCNP: Routing Study Guide.
Author: Todd Lammle.
- [4] Title: CCNP Complete Study Guide.
Author: Wade Edwards.
- [5] Title: Cisco IP Communications Express: CallManager Express with Cisco Unity Express (Networking Technology).
Author: Danelle Au, Baldwin Choi, Rajesh Haridas, Christina Hattingh, Ravi Koulagi, Mike Tasker, Lillian Xia.
- [6] Title: Voice over IP Fundamentals (2nd Edition).
Author: Jonathan Davidson, James Peters, Manoj Bhatia, and Satish Kalidindi.
- [7] Title: Cisco IP Telephony: Planning, Design, Implementation, Operation, and Optimization (Networking Technology). ISBN : 1-58705-157-5
Author: Ramesh Kaza and Salman Asadullah.
- [8] Title: Troubleshooting Cisco IP Telephony.
Author: Giralt, Addis Hallmark, and Anne Smith.
- [9] Title: Administring Cisco QoS in IP Networks . ISBN: 1-928994-21-0
Author: Benolt Durand, Jerry Sommerville, Mark Buchmann , Ron Fuller.
- [10] Title: Voice over IP versus Voice over Frame Relay.
Author: Pauline P. Francis-Cobley and Adrian D. Coward.
- [11] Title: Configuring Cisco Voice over IP. [Electronic Resource]. 2nd Edition. ISBN: 1931836647
Author: Fong, Paul J.
- [12] Title: Cisco IP Routing Protocols. 1st Edition. ISBN: 1584503416
Author: Anand, Vijay.

- [13] Title: Cisco Voice Gateways and Gatekeepers. ISBN: 9781587052583
Author: Donohue, Denise.
- [14] Author: White, Russ. Practical BGP [Electronic Resource]. ISBN: 0321127005123
- [15] Title: Cisco Call Manager Express [Electronic Resource]
Author: Christina Hattingh.
- [16] Title: Cisco IOS in a Nutshell.
Author: James Boney.
- [17] Title: Cisco IOS™ 12.0 Quality of Service. ISBN-10: 1-57870-161-9
Author: Cisco Systems, Inc.
- [18] Title: Cisco® Field Manual: Router Configuration. ISBN-10: 1-58705-024-2
Author: Dave Hucaby, Steve McQuerry.
- [19] Cisco Documentation – Configuring Generic Traffic Shaping.
- [20] Cisco Documentation – Configuring Class Based Traffic Shaping.
- [21] Cisco Documentation – Configuring Frame Relay Traffic Shaping.
- [22] Cisco Documentation – Configuring the MQS with Command-Line Interface.
- [23] Cisco Documentation – Setting Up a Cisco CME System.
- [25] Online Resources :
<http://en.wikipedia.org>
<http://www.google.ca>
<http://proquest.safaribooksonline.com>
<http://dictionary.reference.com>
<http://www.cisco.com>
<http://www.net130.com>
<http://searchnetworking.techtarget.com>

*T*HANKS