Centro Universitário de Brasília UniCEUB
Faculdade de Ciências Exatas e Tecnologia FAET
Curso de Engenharia de Computação

DAIANE VAZ LIMA

ANÁLISE EM REDES MPLS

Brasília – DF
2007

DAIANE VAZ LIMA

ANÁLISE EM REDES MPLS

Trabalho apresentado à banca examinadora da Faculdade de Ciências Exatas e Tecnológicas, para conclusão do curso de Engenharia da Computação. Prof. Orientador: M.Sc. Antonio José Gonçalves Pinto.

Brasília- DF
2007

DAIANE VAZ LIMA

ANÁLISE EM REDES MPLS

COMISSÃO EXAMINADORA

_____

Prof. Orientador Antonio José Gonçalves Pinto

_____

Prof. Francisco Javier de Obaldia

_____

Prof. Marcello Sandi Pinheiro

_____

Brasília, 27 de maio de 2007.

AGRADECIMENTOS

Um agradecimento especial à minha mãe, minha irmã e ao meu esposo Sérgio, pela compreensão e carinho que foram fundamentais para realização deste trabalho.

Agradeço também aos meus professores e colegas de trabalho pela amizade, força e apoio que proporcionaram durante todos estes anos. Em especial, agradeço a Deus pela força e sabedoria.

“Até aqui, nos ajudou o Senhor”

RESUMO

A arquitetura *MultiProtocol Label Switching* (MPLS) possibilita a organização, de forma efetiva, de diversos rótulos através do método de fila *(FIFO)*. Outra característica apresentada no encaminhamento realizado nos roteadores de borda, é a utilização de classes de pilhas de rótulos (*LIFO*) o que permite uma hierarquia de rótulos na pilha e a utilização da técnica de comutação de pacotes.

Um dos principais atrativos da tecnologia MPLS, é proporcionar novos tipos de serviços, além dos encontrados no roteamento convencional. De comutação TCP/IP. Tendo em vista essa possibilidade, a proposta deste projeto tem como finalidade simular uma rede baseada no backbone da RnP, onde será analisada e avaliada seu desempenho, quando usada em rede IP convencional em comutação TCP/IP e quando usada em MPLS.

Nesta simulação serão utilizados, além do software *Network Simulator*, o sistema operacional *Linux*, a linguagem de programação *Otcl* e o *Gnuplot* para plotagem dos gráficos. Serão definidos os agentes de transporte, os protocolos, e o tipo de tráfego injetado na rede, que possibilitará a avaliação de uma tecnologia de alta disponibilidade, como é o MPLS.

Palavras-chaves: MPLS, IP, Simulação, Qualidade de Serviço (QoS)

ABSTRACT

Architecture MPLS makes possible the organization, of form accomplishes, diverse labels through the First method in First out (FIFO). The main characteristic of the classrooms of stacks of labels is the guiding carried through in the roteadores of edge, allowing a hierarchy of labels in the stack and the use of the technique of commutation of packages.

One of main the attractive ones of technology MPLS, is to provide new types of services beyond the found ones in the conventional roteamento. In view of this possibility, the proposal of this project has as purpose to simulate a net based on backbone of the RnP, where it will be analyzed and evaluated its performance.

In this simulation they will be used beyond software Network Simulator, the operational system Linux, the programming language Otcl and the Gnuplot for plotagem of the graphs. The agents of transport, the protocols, and the type of traffic injected in the net will be defined, that will make possible the evaluation of a technology of high availability.

Word-keys: MPLS, Quality of Service (QoS), IP, Simulation

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIAÇÕES e SÍMBOLOS

ARPA — *Advanced Research Projects Agency*

ATM — *Asynchronous Trans fer Mode*

BA — *Behavíour aggregate*

Best Effort — *Melhor Esforço*

CBQ — *Class Based Queuing*

CBR — *Constant Bit Rate*

CQ — *Custom Queuing*

DoD — *Department of Defense*

DSCP — *Dífferentíated Servíces Code Point*

Duplex — *Transmissão em dois sentidos*

EF — *Expedited Forwarding*

FEC — *Forwardíng Equivalency Class*

FIFO — *First In, First Out*

FTP — *File Transfer Protocol*

FTN — *FEC-to-NHLFE*

Full-duplex — *Transmissão de envio e recepção simultâneos*

Hal-duplex — *Transmissão de envio e recepção em tempos diferentes*

Host — *computador ligado a uma rede física*

IETF — *Internet Engineering Task Force*

IHL — *Internet Header Length*

ILM — *Incoming Label Map*

IP — *Internet Protocol*

ISO — *International Organization for Standardizatíon*

ITU — *International Telecommunicatíons Union*

Jitter — *Variação do Atraso*

LAN — *Local Area Network*

LDP — *Label Dístributíon Protocol*

LER — *Label Edge Router*

LSP — *Label Switch Patch*

LSR — *Label Switch Router*

Mbps — *Mega Bits por Segundo*

MF — *Mult Field*

MPLS — *Multiprotocol Label Switch*

NFS — *National Science Foundation*

NHLFE — *Next Hop Label Forwarding Entry*

NS — *Network Simulator*

OSI— *Open System Interconnect*

QoS — *Quality of Services*

QoSR — *Roteamento com Quality of Services*

RFC — *Request For Comments*

RIP — *Routing Information Protocol*

Router — *Roteador*

RSVP — *Resource Reservation Protocol*

Simplex — *Transmissão em apenas um sentido*

SLA — *Service Level Agreement*

SFQ — *Stohastic Fair Quering*

TCP/IP — *Transmission Control Protocol / Internet Protocol*

TE — *Engenharia de Tráfego*

Trace — Arquivo Registro Dados

UDP — *User Datagram Protocol*

VBR — *Variable Bit Rate*

CAPÍTULO 1. INTRODUÇÃO

1.1 MOTIVAÇÃO

Desde o aparecimento da Internet, com toda sua interatividade, as aplicações mais críticas como áudio e vídeo, vêm tendo uma maior divulgação devido ao interesse em pesquisas científicas, seguidas das acadêmicas e atualmente das comerciais. [OLIVEIRA, 2006].

Com isso nasceu também a necessidade em se trabalhar com serviços qualitativos. As aplicações multimídia, por exemplo, vêm sendo cada vez mais utilizadas, tornando o tráfego nas redes muitas vezes maior.

Além dos serviços como *E-mail*, por exemplo, surgiram também outros mais complexos como videoconferência, educação à distância, transmissão de TV e rádio, vídeo em tempo real, telemedicina, vídeo sob demanda, voz sobre IP, comércio eletrônico, entre outros.

Por demandarem requisitos especiais de qualidade de serviço, os quais estão relacionados a aspectos como o desempenho, a confiabilidade e a segurança, estas novas aplicações exigem, para seu bom funcionamento, garantias como simplicidade, escalabilidade e gerenciabilidade. Essas garantias, aliadas ao crescimento em progressão geométrica do tráfego e do número de usuários foram as que alavancaram essa necessidade, tornando mais difícil o uso do roteamento tradicional. [TANENBAUM, 2003].

A Internet oferece um tipo de serviço, chamado de “melhor esforço” (*Best Effort*), onde todos os pacotes de dados recebem o mesmo tipo de tratamento e a rede faz o seu “melhor esforço” para encaminhar todos os pacotes de dados o mais rápido possível. Este tráfego é caracterizado pela falta de garantia na entrega dos pacotes de dados, uma vez que todos recebem o mesmo tipo de tratamento, independente dos requisitos das aplicações.

Em casos de congestionamento de tráfego, pacotes podem ser perdidos ou sofrer atrasos indefinidos e indeterminados. Dessa forma, o usuário não recebe garantia de alguns parâmetros de Qualidade de Serviço (QoS) como largura de banda, atraso e variação do atraso para as suas aplicações.

Considerando tudo isso, as redes deixam de atender, paulatinamente, todas as necessidades dos seus usuários.

Surgiu então, a necessidade de agilizar o processo de roteamento para poder suportar tal tráfego, diferenciando os diversos tipos de fluxos e tratá-los de forma distinta, dando prioridades às aplicações mais críticas.

Na internet existem diferentes tipos de modelos propostos pelo IETF (*Internet Engineering Task Force*) para oferecer Qualidade de Serviços em uma rede TCP/IP (TCP – *Transmission Control Protocol* e IP - Internet Protocol) que se adequam de acordo com o tipo de aplicação e arquitetura da rede. Entre as soluções propostas pelo IETF está o *Multiprotocol Label Switch – MPLS.*

Este trabalho tratará da avaliação dessa tecnologia objetivando mostrar que seu uso pode levar a um melhor rendimento dos recursos da rede, e atender as necessidades de QoS atuais.

1.2 OBJETIVOS

Este documento concentra-se na comparação entre a rede de computadores que utiliza o roteamento convencional de comutação TCP/IP e aquela que emprega a tecnologia MPLS. O grande crescimento da Internet e o surgimento de novas aplicações, passaram a exigir redes de alta velocidade e que demandam garantia de largura de banda e contra perda de pacotes, retardo e variação de atraso (jitter). [SOARES, 1995].

Com a implantação de regras que asseguram o serviço, é possível oferecer uma maior garantia e segurança para as aplicações que demandam prioridade sobre outras menos exigentes. O modelo proposto pela IETF empregando *Multiprotocol Label Switch – MPLS* possibilita o oferecimento de diferentes níveis de QoS exigidos por aplicações mais complexas e trabalha em conformidade com a técnica de “melhor esforço”.[SOARES, 1995]

Os objetivos deste trabalho são o estudo do roteamento convencional de comutação IP, das técnicas de Qualidade de Serviço e do *Multiprotocol Label Switch – MPLS,* visando a uma análise e avaliação comparativa do desempenho entre as duas redes.

A realização de estudos, implementação de serviços de QoS e avaliação de desempenho, apresentam muitas dificuldades, entre elas, a grande

complexidade. Desta forma, será utilizada técnica de simulação, a qual já é aplicada com freqüência, principalmente pela flexibilidade que possibilita testar cenários variados, incluindo o comportamento de vários protocolos, serviços de QoS, e efeito de diferentes topologias de rede.

O ambiente de simulação utilizado para as avaliações foi o *Network Simulator* (ns-2) versão 2.29 [COUTINHO, 2003]. O *Network Simulator* é um simulador orientado a eventos discretos, utilizado em pesquisas de redes de comunicação.

1.3 ORGANIZAÇÃO DO PROJETO

O presente documento está segmentado da seguinte maneira:

- Capítulo 1, de caráter introdutório - trata da caracterização do projeto de pesquisa apresentando o tema, sua motivação e seus objetivos;
- Capítulo 2, serão revisados algumas tecnologias e serviços utilizados no projeto;
- Capítulo 3, é realizada a descrição da metodologia utilizada na pesquisa, além do detalhamento das ferramentas utilizadas no projeto;
- Capítulo 4, apresenta a implementação das simulações do projeto;
- Capítulo 5 faz uma apresentação da análise dos resultados obtidos com a simulação;
- Capítulo 6 traz as conclusões de modo geral obtidas através da simulação.

CAPÍTULO 2. TCP/IP, QUALIDADE DE SERVIÇO E MPLS

Neste capítulo serão revisadas algumas tecnologias e serviços, como: arquitetura e funcionamento de redes TCP/IP, Qualidade de Serviço (QoS) e Arquitetura MPLS a fim de facilitar a compreensão e desenvolvimento do projeto.

2.1 ARQUITETURA TCP/IP

A arquitetura TCP/IP surgiu da pesquisa administrada pela Agência de Projetos de Pesquisa Avançada de Defesa (DARPA) nos anos 60, patrocinada pelo Departamento de Defesa do governo dos Estados Unidos da América, onde o principal objetivo era manter a comunicação entre os diversos sistemas espalhados pelo mundo, despertando assim interesse por um protocolo que estava sendo desenvolvido e utilizado pelas Universidades para interligação dos seus sistemas computacionais.

A ARPANET - *Advanced Research Projects Agency*, surgiu para atender um requisito de garantir uma rede que permaneceria intacta caso um dos servidores perdesse a conexão e necessitava de um protocolo modelo que assegurasse a confiabilidade da rede, flexibilidade e facilidade de implementação. [SOARES, 1995]. Em meados dos anos 70, esta rede inicial evoluiu, teve seu protocolo principal desenvolvido e transformado na base para o TCP/IP*(Transmition Control Protocol / Internet Protocol).*[TANENBAUM, 2003].

O termo “TCP/IP” refere-se a um conjunto de protocolos de comunicação utilizados para troca de dados entre computadores em ambientes de redes locais ou remotas, projetados com o objetivo de construir interconexões dessas redes.

Dentre as várias organizações e comitês que participaram deste desenvolvimento e divulgação do TCP/IP, podemos destacar IETF - *Internet Engineering Task Force*, cuja principal função atual é a manutenção e apoio aos padrões de Internet e TCP/IP principalmente por meio da série de

documentos *RFC* que descrevem as diversas tecnologias envolvidas e servem de base para outras tecnologias

O TCP/IP inclui uma série de padrões que especificam como os computadores vão se comunicar. Além de ser tolerante a falhas, é um protocolo aberto, simples e indispensável para trabalhar com enlaces de baixa confiabilidade. [TANENBAUM, 2003].

Atualmente quando se menciona TCP/IP, vem de imediato a associação com a Internet garantindo uma alta disponibilidade nas redes, o que ocasionou a sua implementação por vários fabricantes.

O protocolo TCP/IP é constituído por 4 camadas: a camada de interface de rede, a camada de rede, a camada de transporte e a camada de aplicação. A seguir a definição das camadas:

2.1.1 Camada de Aplicação

A camada de aplicação dentro do processo de comunicação é representada pelo usuário final. Essa camada faz a conversão entre os diversos tipos de terminais, controles de operação, mapeamentos de memória para os terminais, controle de transferência de arquivos, e-mail, seleção da disciplina de diálogo e outras facilidades.

2.1.2 Camada de Transporte

Apresenta como função permitir a comunicação fim-a-fim entre aplicações, além de ainda controlar o fluxo de dados, controle de erros, seqüênciação e multiplexação do acesso ao nível inter-rede. Nesta camada são definidos dois protocolos, o TCP (Transmission Control Protocol), que é orientado à conexão, confiável, e que permite a entrega sem erros de um fluxo e o UDP (User Datagrama Protocol) que é um protocolo sem conexão, não confiável, para aplicações que não necessitam de controle de fluxo, e manutenção da seqüência das mensagens enviadas.

2.1.3 Camada de Rede

Responsabiliza-se pelo roteamento de dados e é conhecida como nível inter-rede utilizada para atribuir endereço de rede ao sistema e rotear a informação para a rede correta.

2.1.4 Camada de Interface de Rede

A camada de Interface de Rede ou camada de abstração de hardware tem como principal função a interface do modelo TCP/IP com os diversos tipos de redes (X.25, ATM, FDDI, Ethernet, Token Ring, Frame Relay, sistema de conexão ponto-a-ponto).

A figura 2.1 mostra a organização do modelo TCP/IP:

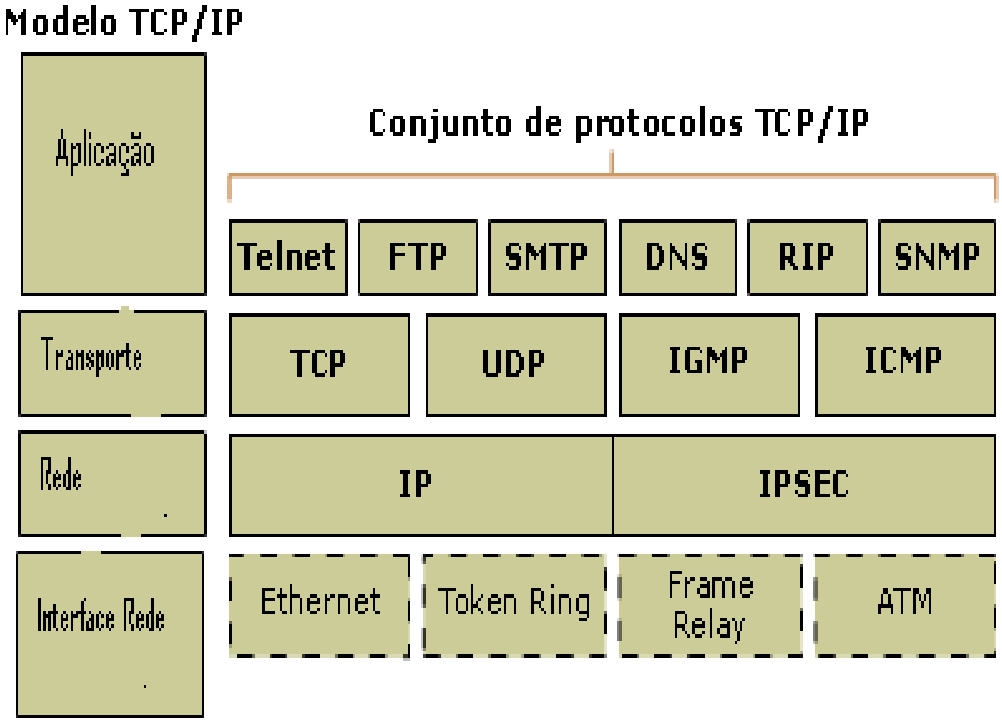

Figura 2. 1 - Modelo TCP/IP
Fonte: TANENBAUM (2003)

2.2 ENDEREÇOS FÍSICOS E LÓGICOS

Cada *host* em uma rede é nomeado com um endereço físico que leva formas diferentes em redes diferentes. Para redes Ethernet, o endereço físico é um valor numérico de 6 (seis) bytes. Cada endereço de host Ethernet é único, e corresponde ao endereço de rede físico instalado no host. Endereços de Internet são endereços lógicos, e são independentes de qualquer hardware particular ou componente de rede. [SOARES, 1995].

Essa numeração lógica é importante ao enviar informação a outros usuários e a outras redes, ou no acesso a máquinas remotas. Esse endereço consiste em um valor numérico de 4 bytes que identifica o número de rede e o número de dispositivo na rede. O endereço IP de 32 bits é representado em notação decimal pontuada onde cada byte representa um valor entre 0 e 255. Computadores em uma rede de TCP/IP possuem nomes lógicos.

Esses nomes lógicos (ou nome de domínio) são associados ao seu endereço IP. Quando um computador deseja comunicar-se com outro, ele precisa traduzir o nome lógico do computador destino (endereço Internet) em seu endereço MAC (endereço físico).

Essa tradução é feita por uma busca em um servidor DNS (Servidor de nome de domínio) que retornará o endereço IP do host de destino. Os endereços IPs são divididos em cinco classes: classe A, B, C, D e E.

2.2.1 Endereçamento Classe A

O primeiro byte especifica a porção de rede; os bytes restantes especificam a porção de host. O bit de ordem mais alto do byte de rede sempre é 0. Os valores de rede 0 e 127 são reservados e há 126 redes classe A.

Existem mais de 16 milhões de valores de hosts para cada rede classe A. O número 127 não é utilizado como rede Classe A porque é um número especial. Por padrão, para a Classe A, foi definida a máscara de sub-rede: 255.0.0.0.

2.2.2 Endereçamento Classe B

Os primeiros dois bytes especificam a porção de rede; os últimos dois bytes especificam a porção de host. Os bits de ordem mais alta 6 e 7 da porção de rede são 10.

Há mais de 16 mil redes classe B e existem 65 mil nodos em cada rede class B. Para a Classe B, foi definida a máscara de sub-rede: 255.255.0.0.

2.2.3 Endereçamento Classe C

Os primeiros três bytes especificam a porção de rede; o último byte especifica a porção de host. Os bits de ordem mais alta 5, 6 e 7 da porção de rede são 110; Há mais de 2 milhões de redes classe C e há 254 hosts em cada rede classe C .

2.2.4 Endereçamento Classe D

Esta classe foi definida com os quatro primeiros bits do número IP como sendo iguais a 1, 1, 1 e 0. A classe D é uma classe especial, reservada para os chamados endereços de Multicast. [ROSEN, 1997].

2.2.5 Endereçamento Classe E

A classe E foi padronizada com os quatro primeiros bits do número IP como sendo iguais a 1, 1, 1 e 1. A classe E também é uma classe especial e está reservada para uso futuro.

2.3 PROTOCOLO IP

O IP (Protocol Internet) é um protocolo não confiável que não garante a entrega dos pacotes, não possui controle de seqüenciamento e não informa ao transmissor a ocorrência de erros. Cada pacote é tratado de forma independente sendo descartado quando são esgotados os recursos da rede.

Os datagramas IPs são fragmentados para se adequarem ao MTU (Maximum Transfer Unit) do hardware. O cabeçalho IP é ilustrado pela figura 2.2 [SOARES, 1995].

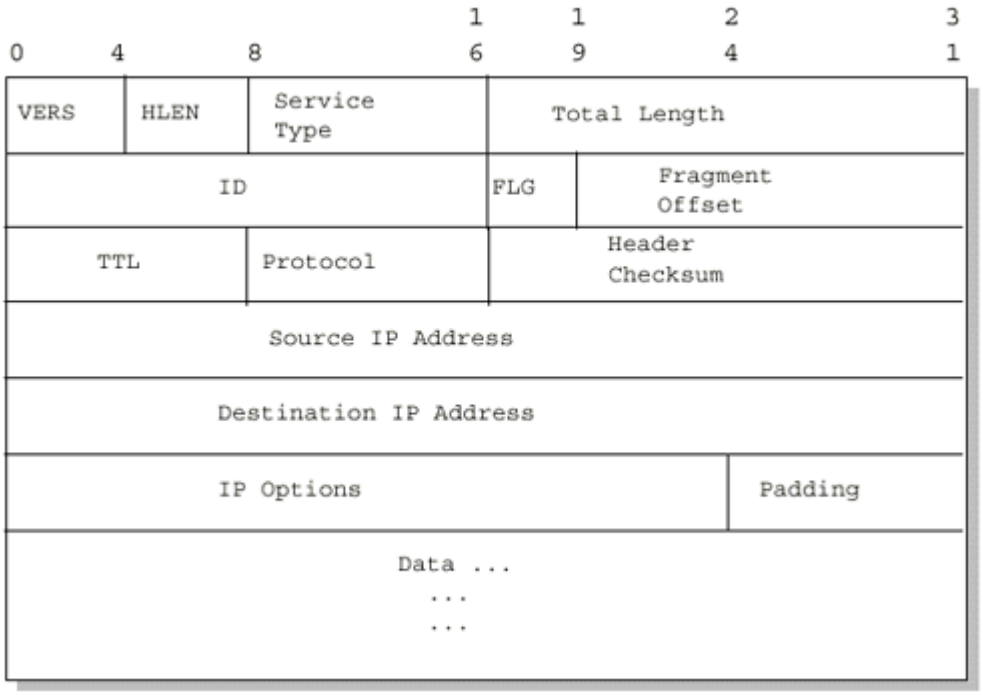

Figura 2. 2 - Cabeçalho IP
Fonte: TANENBAUM (2003)

2.4 PROTOCOLO TCP - CONTROLE DE TRANSMISSÃO

O TCP (Transmission Control Protocol) é um protocolo padrão orientado a conexão que fornece um serviço de entrega de pacotes confiável. Recebe a definição de “Orientado a Conexão” porque todos os aplicativos baseados em TCP como protocolo de transporte precisam estabelecer uma conexão antes de iniciar a troca de dados.

Para estabelecer uma conexão é criado um circuito virtual entre as duas aplicações, e é enviado um fluxo de bytes ao destino exatamente na mesma ordem como eles deixaram a fonte. Antes de iniciar a transmissão, as aplicações envolvidas na conexão obtêm uma porta TCP. Os buffers do TCP esperam o fluxo de dados até preencher um datagrama para então enviá-lo. O protocolo TCP divide a mensagem em segmentos que são nomeados, cada

qual, com um número subsequente que o receptor final utiliza para assegurar a recepção de todos os segmentos na ordem correta. [MURHAMMER, 2000].

Além de dividir a mensagem em segmentos, o TCP também é responsável por retransmitir os segmentos em caso de perda e reagrupá-los no destino. [TANENBAUM, 2003].

Os segmentos TCP são encapsulados e enviados em datagramas IP conforme mostrado na figura 2.3:

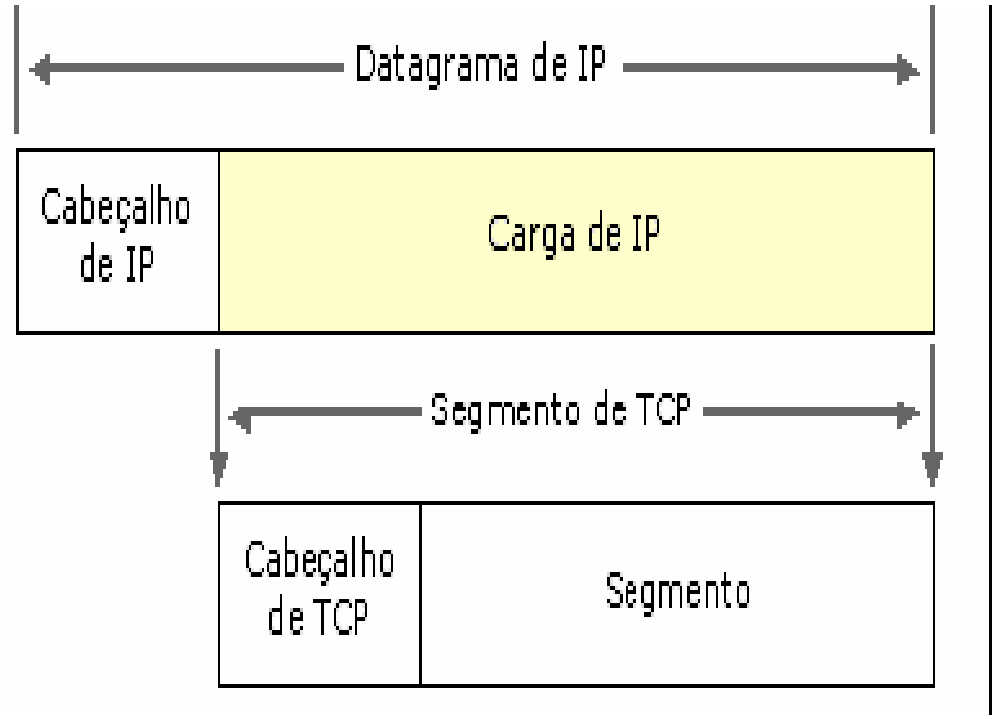

Figura 2. 3 - Encapsulamento TCP
Fonte: TANENBAUM (2003)

2.5 PROTOCOLO UDP - PARA USO DO DATAGRAMA

O UDP (Protocol User Datagram) é um protocolo não orientado à conexão que não faz tratamento de erros, ele transfere essa tarefa para a aplicação.

O UDP fornece um serviço de pacotes sem conexão que oferece a entrega de pacotes com base no melhor esforço, não fornecendo garantia de entrega nem verificação de seqüência de dados. Por não fazer verificações e por não estabelecer sessões, o UDP é um protocolo bastante rápido para a transmissão de dados. [TANENBAUM, 2003].

As mensagens UDPs são também encapsuladas e enviadas em datagramas IP como mostrado na figura 2.4

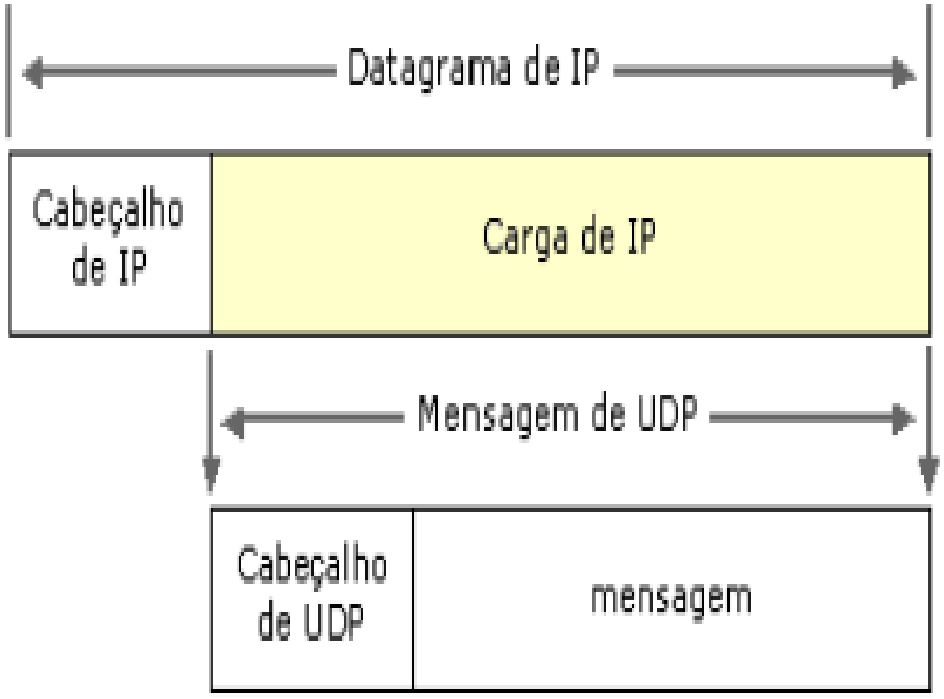

Figura 2. 4 - Encapsulamento UDP
Fonte: TANENBAUM (2003)

2.6 QUALIDADE DE SERVIÇO EM REDES

2.6.1 DEFINIÇÃO

A Qualidade de Serviço ou simplesmente QoS é definida como um efeito de todo desempenho de um determinado serviço, que gera o grau de satisfação de um usuário. Em sistemas multimídias distribuídos, a qualidade de serviço pode ser definida como a representação do conjunto de características qualitativas e quantitativas de um sistema, necessário para alcançar a funcionalidade de uma aplicação. [FAUCHER, 2000].

Em redes de computadores, QoS é utilizado para definir o desempenho de uma rede relativa às necessidades das aplicações, como também o conjunto de tecnologias que permite às redes oferecer garantias de qualidade, desempenho e eficiência.

A QoS pode ser classificada segundo o nível de garantia oferecido:

- QoS rígido (baseado em reserva de recursos) que oferece garantias de forma individual para cada fluxo.
- QoS flexível(baseada em priorização), onde as garantias são oferecidas à grupos, ou agregações de fluxos.

2.6.2 CLASSIFICAÇÃO DE QoS

As abordagens aferidas para QoS são divididas em duas dimensões ou dois escopos, pois determina os limites do serviço de QoS: Tais escopos podem ser classificados em:

- escopo fim- a- fim, quando está acessível para as aplicações nos sistemas finais;
- escopo intermediário, quando os sistemas finais não requisitam diretamente a QoS que necessitam, mas são atendidos por algum elemento de rede habilitado para tal tarefa.

As características relacionadas ao gerenciamento das requisições de QoS são definidas através do modelo de controle. [SILVA, 2004]. São elas:

- Duração: as requisições de QoS podem variar com relação à duração dos níveis de QoS solicitados (minutos, horas, dias, semanas, meses).
- Granularidade: um pedido pode ser para um único fluxo ou para uma agregação de fluxos.
- Local de controle: independente do escopo de QoS, um pedido pode ser controlado pelo sistema final, ou por algum sistema intermediário.

2.6.3 PARÂMETROS DE QUALIDADE DE SERVIÇO

A obtenção de QoS pode ser constatada em um serviço, se todos os parâmetros pré-definidos forem atendidos, de forma que o mínimo seja alcançado. [SILVA, 2004]. Os parâmetros escolhidos para realização deste projeto foram atraso, vazão e descarte de pacotes.

2.6.3.1 VAZÃO

A taxa de transmissão, ou vazão, corresponde à quantidade de informação que pode ser transferido de um ponto a outro de uma rede em um determinado período de tempo.

A transmissão dessas informações esta limitada pelo tamanho da largura de banda. Largura de banda é quantidade de informações que podem ser enviadas através de uma conexão.

2.6.3.2 ATRASO

É o tempo necessário que um pacote leva para sair da origem e ser transmitido até o seu destino. Na tecnologia VoIP, a demora na chegada dos pacotes pode ocasionar um certo desconforto, semelhante ao encontrado nas comunicações utilizando satélites, conforme ilustrado na figura 2.5.

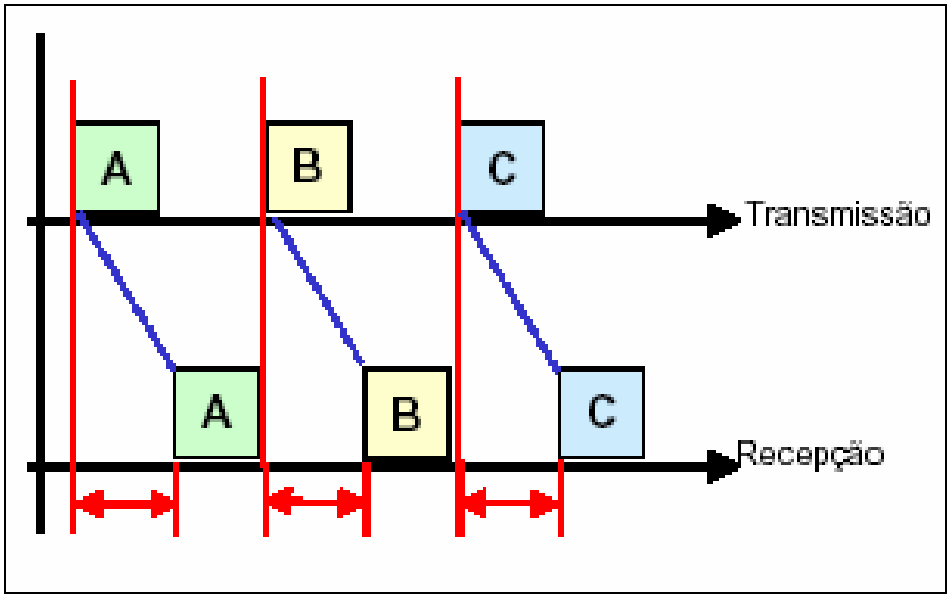

Figura 2. 5 - Atraso de pacote
Fonte: SOARES (2000).

2.6.3.3 PERDAS DE PACOTE

A perda de pacotes é um parâmetro que também merece atenção, pois traz como conseqüência a perda de qualidade na conversação entre os usuários. A conversa se torna “quebrada”, pois os pacotes que se perderam

traziam alguns trechos da conversa. Abaixo serão citadas algumas causas da perda de pacotes.

• Erros: Os pacotes podem ser enviados ou chegarem ao seu destino com algum problema, ou podem ser corrompidos durante a sua transmissão;

• Problemas no enlace: Pode ocorrer algum problema nos equipamentos físicos ou nos meios de transmissões entre os terminais;

• Atraso excessivo: Os roteadores podem descartar pacotes caso o tempo de transmissão seja superior ao tolerável;

• Congestionamento: Os roteadores podem descartar pacotes caso o buffer atinja o máximo permitido;

A combinação de algumas métricas referentes ao Qos, fornece garantia de transmissão, conforme especificado pela norma ITU-T. A tabela 2.1 apresenta as métricas regulamentadas na norma sobredita para tráfego de áudio e vídeo.

Tabela 2. 1 - Norma ITU-T

ITU-T *(Telecommunication Standardization Sector of International Telecomunication Union)*			
Áudio		**Vídeo**	
Vazão (Mbps)	0.1 - mínimo	Vazão (Mbps)	1 - mínimo
Atraso (ms)	150 - máximo	Atraso (ms)	150 - máximo
Descarte (%)	1 - máximo	Descarte (%)	1 - máximo

Fonte: ITU-T.

2.6.4 TECNOLOGIAS DE SUPORTE A QoS

Existem várias tecnologias para provimento de QoS em redes de computadores. O modelo de QoS do ATM, *RSVP* (*Resource reSerVation Protocol*), DiffServ (Differentiated Services), IntServ (Integrated Services) e MPLS (*Multi Protocol Label Switching*) são as principais tecnologias com suporte a QoS. A última será brevemente descrita na seção 2.7. [FAUCHER, 2000].

As aplicações dessas tecnologias estão classificadas e agrupadas de acordo com os serviços que irão oferecer: Nos próximos parágrafos serão apresentadas tais classificações.

2.6.4.1 Aplicações de áudio:

Interativas: Os fatores que influenciam sua qualidade são: fidelidade do sinal de voz, eco, o atraso e *jitter* fim-a-fim e baixa perda. Ex: Telefonia sobre IP;

Não Interativas ou com pouca interatividade: Os fatores que influenciam são: o atraso, *jitter* e perda, mas o grau de influência desses fatores depende da implementação das aplicações. Ex: *Streaming* de áudio de qualidade profissional.

2.6.4.2 Aplicações de vídeo:

Interativas: Os fatores que influenciam são: o atraso, *jitter* fim-a-fim e a perda de pacotes, a depender da tecnologia de vídeo utilizada. Ex: Videoconferência.

Não Interativas: os requisitos variam de acordo com a aplicação, por exemplo: um *streaming* de vídeo de qualidade modesta irá requerer banda em torno de 500 Kbps, poderá tolerar perdas e o *jitte*r, a depender da implementação. Mas uma transmissão de TV deverá ter garantia de banda, baixo retardo, baixo *jitter* e baixa perda de dados.

É importante, ao analisar o oferecimento de QoS, observar se o custo de gerenciar os recursos da rede é menor do que o custo de adicionar mais largura de banda em locais congestionados, por exemplo. Um tratamento e avaliação de riscos bem como uma verificação pró ativa, são necessárias principamente a medida que novas tecnologias tornam-se abundante e comum. [TANENBAUM, 1999]. Em consequência, a QoS será obtida automaticamente.

2.6.5 MECANISMOS DE IMPLEMENTAÇÃO DE QoS

Existem alguns mecanismos de implementação de QoS na rede IP, como "*traffic shaping, queueing, policing"*, marcação de tráfego e fragmentação de pacotes de voz que são aplicados ao tráfego e que já estão dentro da rede IP. [ARC CHART]

Objetivando o controle da intensidade de tráfego na rede, os roteadores em redes IP são providos de *buffers* para armazenamento temporário dos pacotes que chegam. Estes são chamadas de filas de pacotes.

O tratamento que é dado a estas filas está diretamente relacionado ao controle dos parâmetros de QoS, o que ameniza algumas situações momentâneas de congestionamento, [FAUCHER, 2000].

Sendo assim, é de fundamental importância a escolha dos mecanismos de implementação, para garantir o nível adequado que uma rede deverá prover para determinada aplicação.

2.7 TECNOLOGIA MPLS

2.7.1 INTRODUÇÃO

MPLS (*Multi Protocol Label Switching*) é uma tecnologia de comutação de pacotes baseada em rótulos (Labels). Essa tecnologia, além de aumentar o desempenho de encaminhamento de pacotes, apresenta uma alternativa de suporte a Qualidade de Serviço, Engenharia de Tráfego e Redes Privadas Virtuais(VPN). [FAUCHER, 2000].

Esperava-se que o ATM dominasse o cenário mundial devido às suas altas velocidades, porém, ele não era compatível com o protocolo IP, mais difundido do mundo. [GRANADO, 1998]. Vários esforços foram feitos para se compatibilizar o ATM com o IP, mas após algum tempo ficou perceptível a interoperabilidade entre redes IP e ATM. Pois se tratava de uma questão complexa e cara para ser resolvida. Desta forma várias tecnologias independentes foram desenvolvidas na tentativa de contornar esse problema.

O IP Switching, da Ipslon , foi uma das primeiras a aparecer. Ele foi projetado para conectar subredes IP através de nuvens ATM . Apesar desta tecnologia estar fortemente conectada à tecnologia ATM, os comutadores ATM são utilizados apenas como elemento de encaminhamento de pacotes, não sendo utilizada toda sua capacidade de sinalização e controle. [ROSEN, 1998]

Na década de 90 surgiu a arquitetura CSR (*Cell Switching Routers*) , similar ao IP Switching e logo em seguida foi desenvolvido o Tag Switching da CISCO, no qual apresentava a maior parte das características do MPLS. Esta tecnologia pretendia ser apresentada de forma transparente em uma rede heterogênea [GRANADO, 1998]. A medida que um pacote de uma camada superior ingressasse em uma nuvem *Tag Switching*, este pacote era rotulado, uma vez que este rótulo era um indicador de uma base de informações (TIB – *Tag Information Base*). Os rótulos eram similares aos campos do cabeçalho ATM (VPI/VCI), o que facilitava o processamento dos pacotes, permitindo que ele fosse realizado por hardware, garantindo dessa forma, maior agilidade ao processo.

A IBM, em 1995 desenvolveu o ARIS, cujo conceito estava relacionado à comutação IP, desempenhando papel ora de router ora de comutador.

Tendo em conta as várias soluções proprietárias desta natureza, em 1997 foi criado o Multi-Protocol Label Switching (MPLS) Working Group do IETF a fim de propor uma padronização e consolidação das tecnologias Tag Switching da CISCO e ARIS da IBM. [GRANADO, 1998].

2.7.2 ARQUITETURA

Na arquitetura MPLS, a estrutura de encaminhamento de pacotes de dados na camada 3 (camada de rede) é dividida em dois componentes básicos: **componente de controle** e **componente de encaminhamento**.

O primeiro tem por função a criação e remoção dos LSP's (*Label Switch Path*) entre os LSR's (*Label Switch Routers*). O segundo é conhecido como componente de encaminhamento, este é responsável pelo suporte à comutação por rótulos.

O componente de controle é composto de três subsistemas:

- Mecanismo de Descoberta, responsável por descobrir adjacências e manter relações ativas com as mesmas (potenciais parceiros LDP);
- Estabelecimento e Manutenção de Sessão, responsável por estabelecer e manter sessões LDP com as adjacências descobertas para dar suporte à comunicação entre parceiros LDP;
- Distribuição e Gerência de Rótulos, responsável pela alocação, requisição e divulgação de rótulos envolvidos na criação e remoção de LSP's.

A gerência de cada subsistema é feita através de um ponto de acesso, onde é disponibilizada a interface através da qual o administrador da rede MPLS pode ajustar parâmetros de configuração do sistema e/ou o operador pode solicitar a criação/remoção de LSP's. [ASHWOOD, 1999].

O componente de encaminhamento é dividido em dois subsistemas. O primeiro adiciona novas funcionalidades ao núcleo enquanto o segundo atualiza os componentes já existentes para suportar MPLS. A comunicação entre o componente de encaminhamento e o componente de controle ocorre via uma interface bem definida através das chamadas de sistema *ioctl* e *socket*. [ASHWOOD, 1999]. A figura 2.6 mostra os componentes de controle e de encaminhamento da arquitetura MPLS.

Figura 2. 6 - Componentes MPLS
Fonte: ASHWOOD (1999)

2.7.3 COMPONENTES BÁSICOS DO MPLS

2.7.3.1 Label (Rótulo)

Rótulos são utilizados em redes geralmente para associar pacotes ou quadros a conexões. Em MPLS, os rótulos servem para associar datagramas IPs a LSPs (*Label Switch Paths*).

As funções essenciais de um rótulo quando associado a uma FEC (Forwarding EquivalenceClass) são:

- Definir o rótulo o qual será atribuido o datagrama no LSR de ingresso;
- Determinar a rota de um LSP nos LSRs de borda (ingresso) e de núcleo;
- Decidir se o LSR é o de egresso.

Além disso, existem ainda, vários métodos utilizados para a criação dos rótulos:

- Método baseado na topologia –um protocolo de distribuição normal utilizado, como é o caso do OSPF e o BGP (Protocolo gateway de borda).
- Método baseado na requisição – usa um processo baseado nas requisições de fluxo de controle de dados como o RSVP (Protocolo de reserva de recursos).
- Método baseado no tráfego – usa a recepção do pacote para produzir a atribuição e distribuição do rótulo.

Assim que o pacote é rotulado, seu encaminhamento ao destinatário é baseado na troca do rótulo, isto é, cada valor do rótulo tem um significado local pertencendo somente aos enlaces entre LSRs. Quando um pacote recebe sua classificação em uma FEC nova ou existente, um rótulo será atribuído ao pacote de dados.

Teoricamente os rótulos devem ser posicionados nos quadros de enlace, entretanto, existem algumas tecnologias (PPP e Ethernet) que não possuem identificadores de conexão na camada 2 e quando possuem VPI (Virtual Path Identifier) e VCI (Virtual Channel Identifier) no ATM e DLCI no Frame Relay) consistem de um número reduzido de bits. Dessa forma, foi proposto pelo IETF que o MPLS deveria utilizar formatos existentes de rótulos. [ASHWOOD, 1999].

Por isso, o MPLS suporta três tipos distintos de rótulos. Em hardware ATM, usa os bem definidos rótulos VCI e VPI. Em Frame Relay, usa o rótulo DLCI (Data Link Connection Identifier) e em qualquer outra infra-estrutura, usa um rótulo genérico conhecido como Shim, que se posiciona entre as camadas 2 e 3 [FAUCHER].

Como o MPLS permite criar novos formatos de rótulos sem a necessidade de trocar os protocolos de roteamento, é permitido estender a tecnologia para formas de transporte óptico emergentes, como DWDM e comutação óptica.

2.7.3.2 LSR (*Label Switch Routers*)

LSRs são os roteadores de comutação por rótulos. Trata-se de equipamentos de comutação (por exemplo: roteadores IP, switches ATM) habilitados para MPLS. Eles devem ter implementados algumas funcionalidades definidas pelo MPLS. São equipamentos situados no núcleo da rede MPLS, e sua função é encaminhar pacotes baseados apenas no rótulo de cada pacote. Ao receber um pacote, cada LSR troca o rótulo existente por outro, passando o pacote para o próximo roteador e assim por diante. [ROSEN, 1997].

2.7.3.3 LER (*Label Edge Routers*)

LERs (Roteadores de Rótulos de Borda) são LSRs situados na periferia da rede MPLS. Eles ligam diversas sub-redes (Ethernet, Frame Relay, ATM) à rede MPLS e são responsáveis pela designação e retirada de rótulos para o tráfego que entra ou sai da rede MPLS.

O LER é um LSR que além das funções de encaminhamento e controle, quando está na entrada de um domínio MPLS, é por atribuir os pacotes a uma classe de equivalência de encaminhamento (FEC). Este processo de ligação de pacotes a uma FEC pode ser tão complexo quanto necessário, sem afetar o desempenho geral da arquitetura, pois é efetuado somente na admissão do pacote.

2.7.3.4 FEC (*Forwarding Equivalency Class*)

Uma FEC consiste em um conjunto de pacotes que serão encaminhados da mesma maneira em uma rede MPLS. Pacotes de um mesmo fluxo de dados geralmente pertencem à mesma FEC. Requisitos de QoS também podem ser definidos com a designação de FECs. A FEC é representada por um rótulo, e cada LSP é associado a uma FEC. Ao receber um pacote, o LER verifica a qual FEC ele pertence e o encaminha através do LSP correspondente. [ROSEN, 1997].

Portanto há uma associação pacote-rótulo-FEC-LSP. A associação pacote- FEC acontece apenas uma vez, quando o pacote entra na rede MPLS. Isto proporciona grande flexibilidade e escalabilidade a este tipo de rede.

Existem dois tipos de elementos FEC:

- Prefixo de rede: no formato IPv4, possui comprimento arbitrário de 0 a 32 bits;
- Endereço de nó: é um endereço IP de classe A, B ou C.

Um elemento FEC pode contemplar ainda informações adicionais sobre:

- A origem do datagrama (endereço IP de origem);
- A carga do datagrama (protocolo e portas de transporte).

A FEC permite que um grupo específico de pacotes, receba um mesmo tratamento da origem ao destino, pois, ao contrário da remessa convencional do IP, em MPLS, a atribuição de um determinado pacote a uma classe é baseada no prefixo do endereço.

A figura 2.7 ilustra o princípio de funcionamento da FEC.

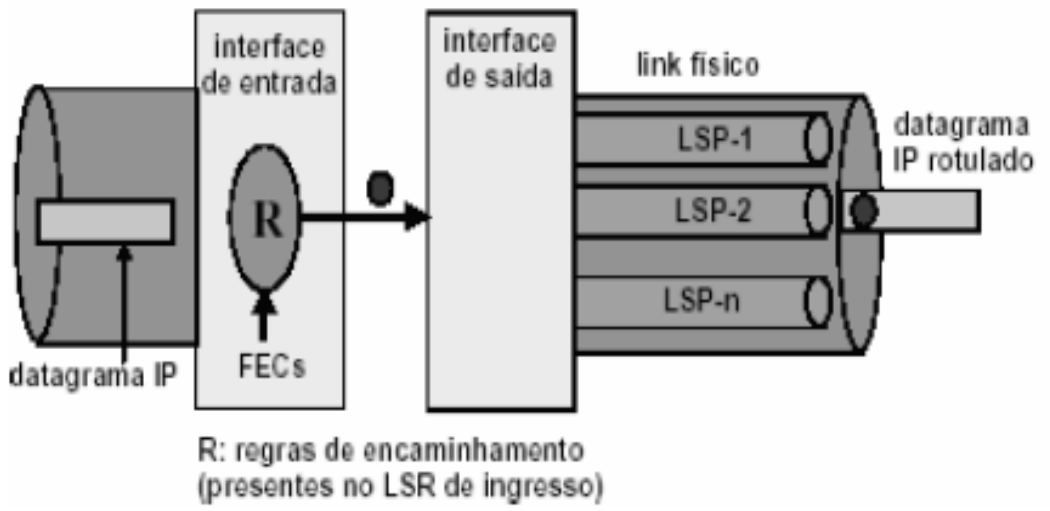

Figura 2. 7 - Funcionamento da FEC
Fonte: ASHWOOD (1999)

2.7.3.5 LSP (*Label Switch Path*)

LSP consiste em um caminho comutado por rótulo, ou seja, um caminho através de uma seqüência ordenada de LSRs, estabelecido entre uma origem e um destino. Um LSP é unidirecional, portanto é preciso ter dois LSPs para uma comunicação entre duas entidades.[OLIVEIRA].

Um LSP é um caminho através do qual transitarão pacotes de uma mesma classe, que compartilham o mesmo destino. Assim, uma rota deve ser estabelecida inicialmente. Isto é feito através de protocolos de roteamento convencional.

Então o caminho fica definido e os pacotes pertencentes a ele não precisam mais ser roteados. Eles serão apenas comutados com base nos seus rótulos. Estes rótulos são distribuídos entre LSRs no momento do estabelecimento de LSPs. [ROSEN, 1997].

O LSP é o componente responsável por determinar a distribuição das rotas antes da transmissão. Quanto à forma de distribuição dos trajetos o LSP possibilita:

- Distribuição hop-by-hop: Cada LSR seleciona, independentemente do próximo hop (pulo) do pacote, aqueles pertencentes a uma FEC

determinada. Esta metodologia é similar às usadas em redes IP como BGP e OSPF

- Distribuição explícita: Nesta distribuição o LSR de ingresso determina a lista dos nós em que o pacote irá passar. Seu trajeto determinado pode assegurar qualidade de serviço no tráfego dos dados, possibilitando uma otimização do fluxo de toda a rede. O estabelecimento do LSP para uma FEC é unidirecional e o seu retorno necessita de outro LSP.

2.7.3.6 Label Stack (pilha de rótulos)

A comutação de rótulos foi projetada para ser usada em redes de grande porte, e o MPLS suporta comutação de rótulos com operações hierárquicas, baseadas na habilidade de o pacote carregar mais de um rótulo. O empilhamento de rótulos permite que LSRs designados troquem informações entre si e ajam como nós de borda para um grande domínio de redes e outros LSRs. [ROSEN, 1997].

Estes outros LSRs são nós internos ao domínio, e não se preocupam com rotas inter-domínio, nem com os rótulos associados a essas rotas. O processamento de um pacote rotulado é completamente independente do nível de hierarquia, ou seja, o nível do rótulo é irrelevante para o LSR. O processamento é sempre baseado no rótulo da pilha, abstraindo-se dos outros rótulos que podem haver abaixo deste.

2.7.3.7 LDP (*Label Distribution Protocol*)

A arquitetura MPLS não define um único método de distribuição de rótulos. Os protocolos já existentes, como BGP (*Border Gateway Protocol*) ou RSVP podem ser estendidos. Pode também ser usado o LDP, definido pelo IETF. [OLIVEIRA]

O LDP é uma especificação que permite a um LSR (Roteador de Comutação de Rótulos) distribuir rótulos. Quando um LSR atribui um rótulo a uma FEC (Classe de Equivalência de Envio), é preciso que ele deixe que seus pares saibam desse rótulo e seu significado. [ROSEN, 1997].

O LDP é usado para este propósito. Já que um conjunto de rótulos do LSR de entrada ao LSR de saída em um domínio MPLS define um caminho de comutação de rótulos (LSP), e já que rótulos são mapeamentos de roteamento da camada de rede para a os caminhos comutados da camada de enlace, o LDP ajuda no estabelecimento de um LSP através do uso de um conjunto de procedimentos para distribuir os rótulos entre os LSR.

LSRs utilizam rótulos para encaminhar tráfego. Um passo fundamental para a comutação de rótulos é que LSRs concordem em relação a quais rótulos eles devem usar para encaminhar o tráfego.[OLIVEIRA]. Eles chegam a este entendimento utilizando o LDP. LDP é uma das partes mais importantes do MPLS. Mecanismos similares para troca de rótulos existiram em implementações proprietárias como o IFMP (*Ipsilon's Flow Management Protocol*), ARIS (*Aggregate Route-based IP Switching*, da IBM), e o *Tag Distribution Protocol* da Cisco. LDP e rótulos são a base da comutação de rótulos.

O LDP possui as seguintes características básicas:

- Oferece um mecanismo de "descoberta" de LSR para permitir que LSRs encontrem uns aos outros e estabeleçam comunicação;
- Define quatro classes de mensagens: *DISCOVERY, ADJACENCY, LABEL, ADVERTISEMENT*, e *NOTIFICATION* (mensagens de notificação);
- Ele roda sobre TCP para proporcionar fidelidade de mensagens.

2.7.3.8 LIB (Tabela para Encaminhamento)

As LIBs são tabelas responsáveis pelo processo de encaminhamento de pacotes e são mantidas pelos LSRs. Elas consistem basicamente de um campo de índice que é preenchido pelo valor do rótulo, uma ou mais entradas, contendo o rótulo de saída, interface de saída e endereço IP do próximo salto (next hop address).

A figura 2.8 ilustra os componentes básicos da arquitetura MPLS.

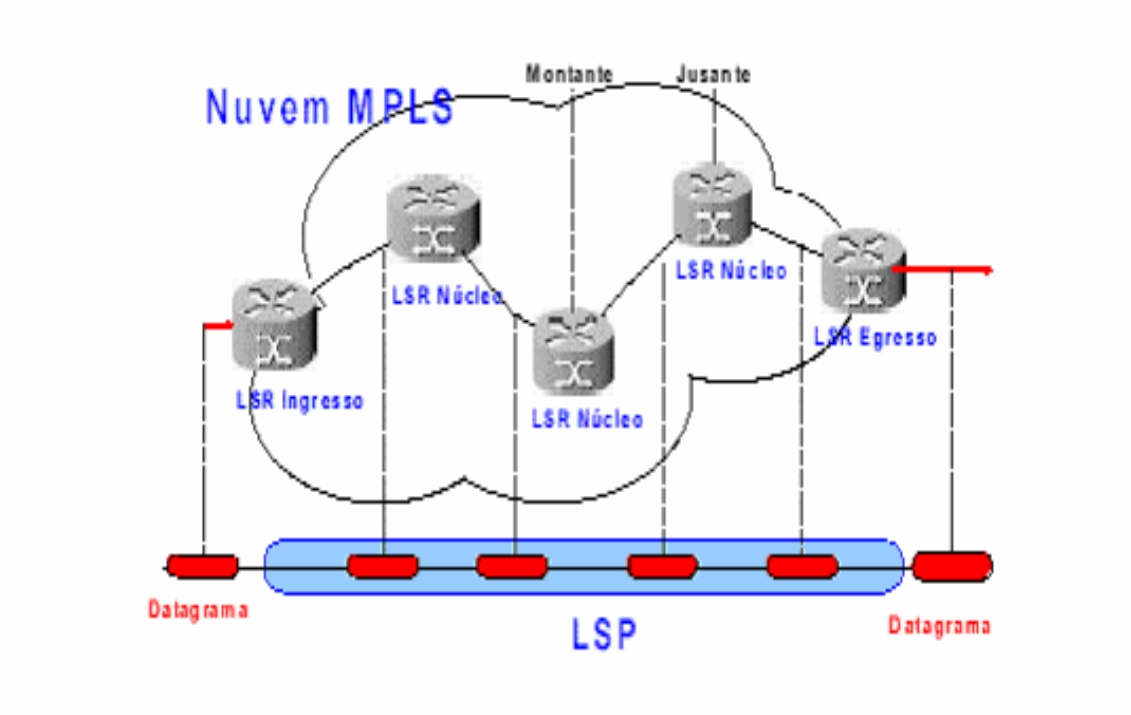

Figura 2. 8 - Componentes Básicos MPLS.

Fonte: FAUCHER (2000)

2.8 FUNCIONAMENTO DO MPLS

Quando um pacote IP entra numa rede MPLS o LSR de igresso, irá associá-lo a uma FEC, caso já exista uma FEC para este pacote. Caso contrário o LSR de egresso irá criar-lhe uma FEC. Desta forma o pacote receberá um *label* e como a FEC está relacionada a uma LSP, que será encaminhado através desta LSP.

A cada LSR pelo qual o pacote passa, os labels são trocados, pois cada label representa um índice na tabela de encaminhamento do próximo roteador. Sendo assim, quando um pacote rotulado chega, o roteador procura em sua tabela MPLS pelo índice representado pelo label. Ao encontrar este índice o roteador substitui o label de entrada por um label de saída associado à FEC a que pertence o pacote. Após completada a operação de troca de labels o pacote é encaminhado pela interface que está especificada na LIB. Na figura 2.9 é ilustrado o momento em que o pacote chega ao roteador A da rede MPLS, o label é removido e o pacote é encaminhado pela interface associada à

FEC a qual pertence. Neste momento o pacote deixa de ser analisado pelo protocolo MPLS e é roteado normalmente pelos protocolos de roteamento. Chegando ao roteador B de saída. [GRANADO, 1998].

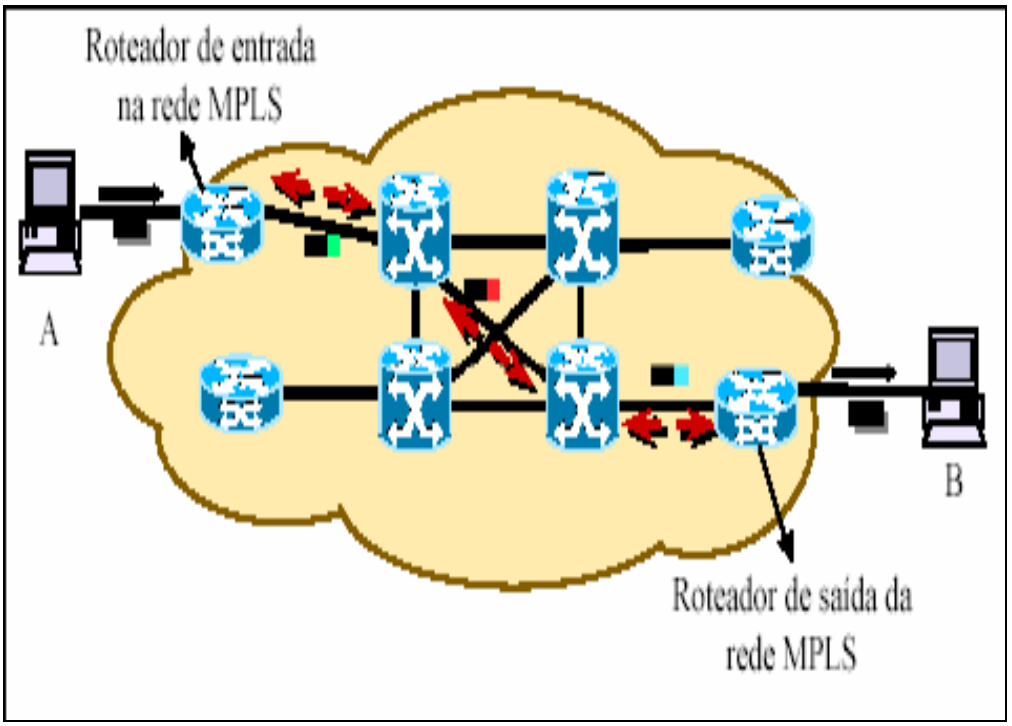

Figura 2. 9 - Funcionamento do MPLS

Fonte: FAUCHER (2000)

2.9 APLICAÇÕES DE MPLS

Uma das principais aplicações para MPLS é a Engenharia de Tráfego, pois permite alterar o caminho normal que alguns pacotes seguiriam no caso de serem encaminhados pelo roteamento IP(convencional), ou seja, pelo caminho mais curto escolhido pelo protocolo de roteamento. [GRANADO, 1998].

O MPLS consegue forçar pacotes a seguirem certas rotas preestabelecidas, o que não acontece na estrutura convencional. O MPLS pode ser utilizado também para facilitar a integração de IP com ATM. Usando roteadores MPLS e fazendo uma atualização dos *switches* ATM permite fácil integração entre ambas as tecnologias. *Switches* ATM podem se comportar como LSRs, fazendo o encaminhamento baseado em rótulos.

O MPLS também permite facilmente a configuração de Redes Privadas Virtuais (VPNs). A VPN é uma rede particular, onde as informações podem trafegar de forma segura. Esta é construída sobre a infra-estrutura de uma rede pública, como a Internet. Na VPN é utilizado o tunelamento, onde os pacotes são transmitidos na rede pública em um túnel privado que simula uma conexão ponto-a-ponto. [GRANADO, 1998]. Desta forma, as VPN's facilitam a extensão de redes corporativas e empresariais a pontos distantes.

A utilização do MPLS como mecanismo de encaminhamento de um domínio VPN provê agilidade, facilidade de gerenciamento para grandes redes, suporte à segurança, suporte a QoS além de possuir como características inerentes a confidencialidade, a integridade e autenticidade dos dados.

CAPÍTULO 3. METODOLOGIA

Este capítulo descreve o ambiente de simulação adotado para a implementação da tecnologia MPLS, mostrando um descritivo dos métodos utilizados e apresentando o planejamento de simulação, tipo de simulação adotada, etapas do trabalho e as ferramentas utilizadas.

As etapas seguidas no desenvolvimento do presente trabalho foram:

1. Estudo geral sobre redes de computadores, englobando conceitos básicos e tecnologias de transmissão de dados;
2. Estudo sobre arquitetura TCP/IP, descrevendo suas pilhas de protocolos e suas funcionalidades;
3. Estudo sobre a tecnologia MPLS, entendendo seu suporte a implementação de QoS e Engenharia de Tráfego e o emprego de simulações, para avaliar suas funcionalidades e desempenho;
4. Simulação da rede IP convencional e MPLS, verificando o uso mais eficiente dos recursos de rede que satisfarão os requisitos das aplicações e o aumento de rendimento da rede. Na abordagem, foram executados experimentos de simulação para medição da vazão, atraso e taxa de perda de pacotes, em várias instantes de um cenário representativo;
5. Estudo sobre simulador de redes Network Simulator (NS-2), por se tratar de uma ferramenta aberta, de amplo uso na comunidade científica que trabalha com diversos tipos de tecnologias de redes;
6. Simulação da rede MPLS, com a inclusão de LSRs dentro da rede IP formando um domínio MPLS.
7. Implementação dos *scripts* necessários para a simulação da rede, definindo as seguintes características:

- protocolos e/ou agentes de transporte para encaminhar os pacotes pela rede;
- fluxos, agregações dos fluxos e tamanho dos pacotes;
- tempo de simulação;
- analise dos resultados da simulação.

3.1 SIMULAÇÃO

Simulação é uma técnica muito utilizada para avaliação de desempenho de sistemas, em sua maioria quando o sistema avaliado não está disponível. No caso de redes de computadores, como a Internet, apesar do sistema já estar implementado e operacional, uma alteração em um sistema crítico pode trazer resultados indesejados além de muitos contratempos.

Assim, a realização de simulações é o melhor método para obter boas estimativas do comportamento do sistema após uma possível alteração de algum protocolo. [COUTINHO, 2003].

Uma simulação baseada em registro de eventos é aquela que apresenta um registro que contém eventos ordenados no tempo, observados em um sistema real, como entrada. Esse tipo de simulação é utilizado na análise de algoritmos de alocação de recursos. Uma característica importante do trace é a credibilidade, pois ele é utilizado como entrada da simulação, a qual pode incluir diferentes algoritmos para serem avaliados sob as mesmas condições de demanda. [COUTINHO, 2003].

Um dos principais problemas dos traces é o tamanho, pois representam seqüências longas e exigem um tempo computacional considerável para serem processados. Simular a Internet representa um grande desafio devido às suas características ímpares, que torna difícil obter um caráter preciso a seu respeito.

A heterogeneidade da internet facilita a inclusão de várias tecnologias diferentes de rede, o que possibilita conexão de todas as redes à ela, causando uma enorme dificuldade em compreender o seu comportamento.

Como ocorrem mudanças drásticas, existe uma grande dificuldade para simular a Internet. Estudos revelam resultados surpreendentes, como crescimento exagerado do tráfego em determinada situação e depois uma diminuição expressiva a níveis semelhantes aos do início do crescimento. [COUTINHO, 2003].

3.2 FERRAMENTAS

Aqui serão mostradas e descritas as ferramentas utilizadas, e as linguagens de programação inerentes ao desenvolvimento do projeto.

3.2.1 SIMULADOR DE REDES *NETWORK SIMULATOR (*NS*)*

O *Network Simulator* (NS) [COUTINHO, 2003] é um simulador baseado em eventos discretos e orientado a objetos para a simulação de redes, resultante de um projeto conhecido como VINT (*Virtual InterNetwork Testbed*). O objetivo do NS é proporcionar um ambiente para o desenvolvimento de pesquisas em torno dos protocolos que constituem a Internet, isto é, que utilizam a pilha TCP/IP, tanto no contexto das redes fixas quanto móveis, com fio e sem fio.

Uma grande vantagem do NS reside no fato de ele ser totalmente gratuito e com código fonte aberto, o que permite ao usuário proceder os ajustes que julgar necessários. [COUTINHO, 2003]. O NS é um dos simuladores mais utilizados atualmente pela comunidade cientifica para pesquisas em rede de computadores, ele provê suporte ao TCP e variantes do protocolo, redes sem fio, MPLS e implementa filas de roteamento do tipo *droptail*, DiffServ RED (*Random Early Detection*), *fair quering* (FQ), *stohastic fair quering* (SFQ), *class-based quering* (CBQ) dentre outras. O NS possui um visualizador gráfico para animações da simulação (NAM – *network animator*), *timers* e escalonadores, conforme ilustra a figura 3.1:

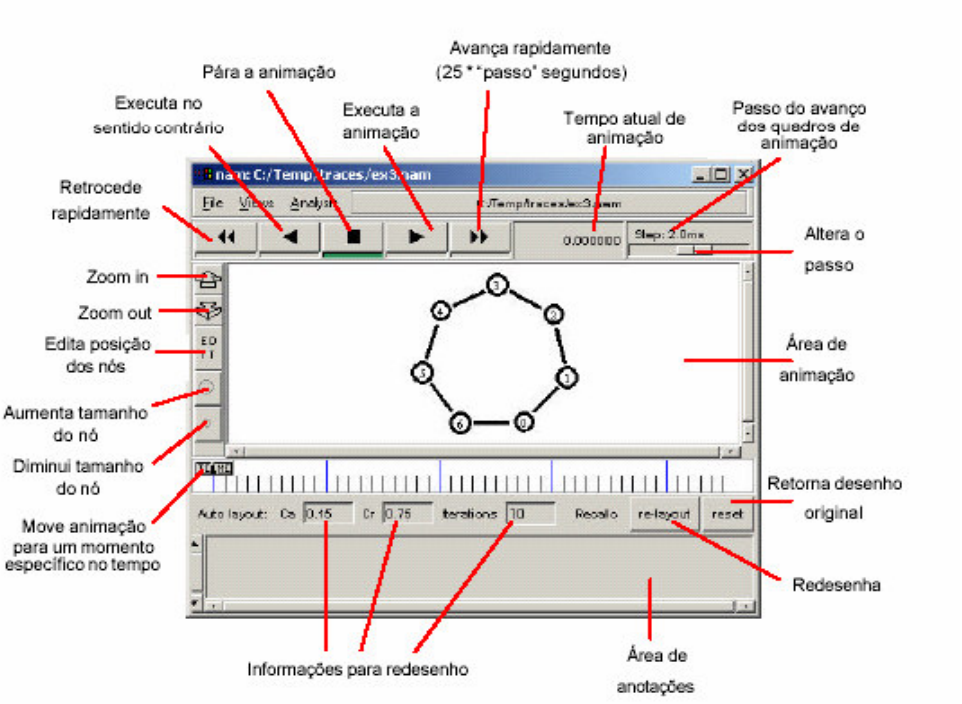

Figura 3. 1 - NAM
Fonte: COUTINHO (2003)

A estrutura do NS foi baseada na linguagem orientada a objetos C++, de forma modular. O uso desta linguagem nos módulos permite velocidade e mais praticidade na implementação de protocolos e modificação de classes. A interface entre o usuário e o NS dá-se através da linguagem *script* OTcl. [COUTINHO, 2003]. A figura 3.2 ilustra a estrutura do NS.

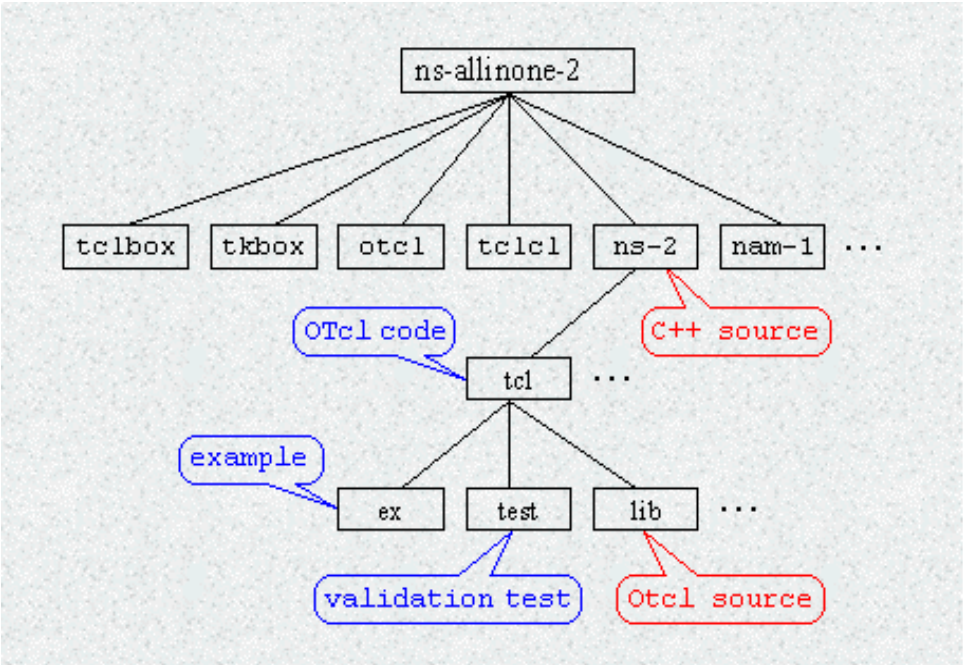

Figura 3. 2 - Estrutura do NS
Fonte: COUTINHO (2003)

A figura 3.3 apresenta os principais componentes utilizados na simulação do projeto aqui proposto com o uso do NS e a interação entre cada um deles.

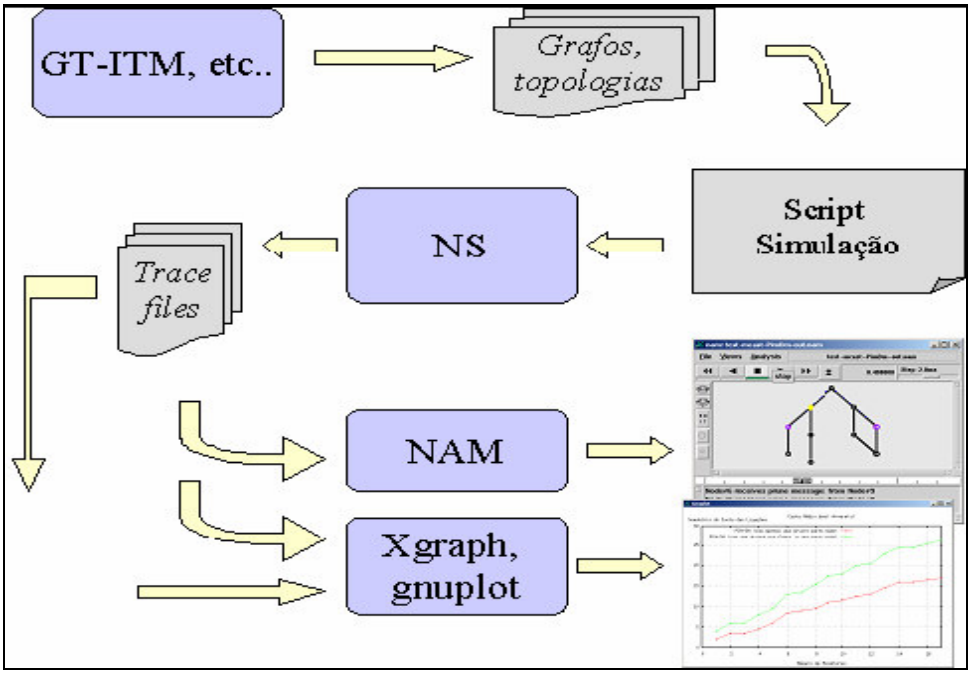

Figura 3. 3 - Arquitertura do NS
Fonte: COUTINHO (2003)

Os escalonadores de eventos e os componentes de rede são implementados em C++ e disponibilizados ao interpretador *OTcl* através de uma replicação feita pela camada *tclcl,* que recria os objetos C++ em objetos OTcl, e que podem finalmente ser manipulados por esta última (processo denominado *linkag*e). Todo o conjunto constitui-se no NS, que é um interpretador de OTcl com bibliotecas de simulação para redes de computadores. [COUTINHO, 2003].

Uma questão importante que deve ser observada, é que o NS não fornece estatísticas de simulação de forma automática. Estas estatísticas devem ser obtidas através de procedimentos matemáticos no *script.* Estes scripts são necessários na formulação da simulação propriamente dita, no qual serão coletados os resultados para análise Pode-se, ainda, usar ferramentas para análise de arquivos de registros gerados durante a simulação, que são os verdadeiros resultados da simulação.

Visualizar a simulação é muito útil para auxiliar a compreensão do cenário simulado. Inclui a visualização da topologia e a animação da simulação, que mostra o tráfego de pacotes.

Em simuladores utilizados pela comunidade científica, os cenários são configurados através de arquivos de entrada e os resultados da simulação são arquivos de saída..

3.2.2 FORMATO DO ARQUIVO TRACE

O arquivo de trace .nam permite a visualização da simulação com a ferramenta Network Animator (NAM). A visualização é muito importante para analisar o tráfego da rede de forma gráfica, baseada em animações. Através do NAM pode ser observada a topologia da rede, a transmissão dos fluxos, o conteúdo do fluxo, a formação de filas, o descarte de pacotes, etc.

Os resultados obtidos com a simulação, através do NS, foram obtidos através da análise do arquivo de traces gerado com registro dos eventos ocorridos durante o processo de simulação. O arquivo de trace, que contém campos como evento, tempo, nó origem, nó destino, entre outros, pode alcançar vários Megabytes de tamanho. O formato padrão do arquivo *trace* é mostrado na figura 3.4.

Evento	Tempo (s)	do nó	para nó	Tipo pacote	Tamanho Pacote (bytes)	flags	fid	Endereço fonte	Endereço destino	Número seqüência	ID do pacote
r	1.3556	0	2	tcp	30	--------	1	0.0	3.0	29	19

Tipos de eventos:
r: recebe
+: *enqueue*
- : *dequeue*
d: descarta

Figura 3. 4 - Formato Trace
Fonte: COUTINHO (2003)

O primeiro campo diz respeito ao evento ocorrido. Pode ser uma entrada em fila (+), uma saída de fila (-), um descarte de pacote (d), um recebimento de pacote (r), etc. O campo seguinte é o momento da simulação no qual o evento ocorreu.

Os dois campos seguintes são referentes ao intervalo de Nós no qual o evento ocorreu. Os dois próximos campos dizem respeito ao tipo (TCP ou

UDP) e tamanho do pacote (em bytes), respectivamente. A seguir tem-se uma série de flags relacionados à notificação antecipada de congestionamento, mas normalmente não são utilizados. Depois tem-se a identificação do fluxo, os endereços do transmissor e do destinatário, o número de seqüência do pacote e, finalmente, um número que identifica de forma única o pacote na rede.

3.2.3 SISTEMA OPERACIONAL LINUX

O *Network Simulator (*NS) foi construído para rodar preferencialmente em plataformas Unix, mas podendo também rodar em plataforma Microsoft Windows, apesar de que sua instalação não é tão automatizada e não segue os padrões de instalação dos softwares feitos para Windows.

O Linux é um sistema operacional livre [GAY, 1999]. Foi escrito por Linus Torvalds com a assistência de um grupo técnico altamente capacitado através da Internet. Possui as mesmas características presentes nos mais modernos sistemas operacionais, incluindo multitarefa real, memória virtual, *shared libraries* (bibliotecas de "linkagem" dinâmica), carregamento de *drivers* sob demanda, suporte nativo a redes TCP/IP, fácil integração com outros sistemas operacionais e padrões de rede, nomes longos de arquivos, proteção de acesso a recursos compartilhados, suporte a vários idiomas e conformidade com os mais respeitados padrões internacionais.

Além de todas as características relacionadas acima o sistema operacional Linux é um software totalmente gratuito e de código aberto, distribuído gratuitamente na Internet, o que viabiliza muito a sua utilização para estudos e pesquisa.

3.2.4 LINGUAGENS DE PROGRAMAÇÃO

È através do script OTcl que se dá a interface entre o usuário e o NS. A divisão em duas linguagens (OTcl e C++) é importante, pois permite ao simulador realizar os testes com maior velocidade, flexibilidade e facilidade de mudança de parâmetros. O núcleo do NS, os escalonadores de evento e os componentes de rede são implementados em C++ e disponibilizados ao

interpretador OTcl através de uma replicação feita pela camada *tclcl,* que recria os objetos C++ em objetos OTcl, e podem ser manipulados através do processo de *linkag*e. O conjunto apresentado, constitui-se no NS, que é um interpretador de Otcl com bibliotecas de simulação para redes de computadores. [COUTINHO, 2003].

Primeiramente, para se montar uma simulação no NS é preciso escrever um *script* em tcl. Este *script* contém algumas partes básicas inerentes ao processo de simulação.

Quando se cria o objeto simulador, os arquivos de registros são abertos. Nesses arquivos são registrados todos os eventos da simulação, possibilitando realizar as análises.

3.2.5 GNUPLOT

O aplicativo gnuplot é destinado à visualização de gráficos e superfícies, úteis em aplicações científicas. Mas precisamente nas áreas de física, matemática, estatística, engenharias (cartográfica, mecânica, elétrica, ...), etc. [GALO, 2003].

Este aplicativo será utilizado na plotagem dos gráficos, a partir do arquivo de registros gerado pelo simulador.

3.3 PROCEDIMENTOS UTILIZADOS NO DESENVOLVIMENTO DO PROJETO

Para simular a rede *MPLS* submetida à aplicações multimídia como fontes de dados, voz e vídeo, e avaliação de seu comprometimento com relação ao oferecimento de Qualidade de Serviço (QoS) foi utilizada a técnica de simulação, através do Simulador de Redes *Network Simulator (NS).*

3.4 DEFINIÇÃO DOS CENÁRIOS

Os cenários definidos neste trabalho de pesquisa foram baseados em:

a) A simulação de uma rede IP conhecida (RNP), o qual apresenta serviço de melhor esforço. Dentro deste cenário serão demonstrados os seguintes aspectos:
    - nós e os enlaces da rede
    - tráfego utilizado na rede
    - protocolos ou agentes de transporte
    - fluxos e tamanho dos pacotes de dados;
    - parâmetros considerados: **ATRASO, VAZÃO e PERDA DE PACOTES**.
b) A simulação, com o NS (Network Simulator), da mesma topologia, utilizando a tecnologia MPLS. Dentro deste cenário serão demonstrado os seguintes aspectos:
    - nós e os enlaces da rede
    - tráfego utilizado na rede
    - protocolos ou agentes de transporte
    - marcação do tráfego a ser priorizado;
    - políticas de admissão, de filas e de encaminhamento de pacotes.
    - parâmetros considerados: **ATRASO, VAZÃO e PERDA DE PACOTES**.

Serão definidos, para realizar a medição dos parâmetros adotados nesta análise, os nós e os enlaces da rede, aplicações, geradores de filas, gerenciadores de filas, a capacidade do link em carregar dados em um sentido ou em ambos simultaneamente, ou seja, será definido todo o tráfego a ser injetado na rede

3.5 PLANEJAMENTO DA SIMULAÇÃO

O planejamento da simulação consiste na definição dos objetivos, das métricas, dos parâmetros e da quantidade de replicações da simulação. Os objetivos da simulação, já definidos no capítulo 1, serão atendidos através dos parâmetros medidos (Vazão, Perda de pacotes e Atraso).

3.5.1 ESCOLHA DE PARÂMETROS

Para realizar a medição dos parâmetros de vazão, atraso e descarte de pacotes, utiliza-se o arquivo de registros gerado na simulação, relacionando-o com o tempo de execução da simulação.

- **Atraso** - É o tempo necessário para um pacote percorrer a rede, medido do momento em que é transmitido pelo emissor até ser recebido pelo receptor.
- **Vazão -** É o montante de tráfego de dados movidos de um nó da rede para outro em um determinado período de tempo. A vazão representa um parâmetro primordial para se obter QoS para aplicações.
- **Perda de pacotes** - A perda de pacotes representa o número de pacotes que foram transmitidos na rede, mas não alcançaram seu destino em um determinado período de tempo.

3.5.2 QUANTIDADE DE REPLICAÇÕES

Definidos os parâmetros foi necessário determinar a seqüência de execuções realizadas na simulação, ou seja, quantas replicações são necessárias para a obtenção de resultados consistentes.

A quantidade baseia-se na amostragem dos dados obtidos e a média entre eles. [GALO, 2003]. O nível de confiança das conclusões depende do tamanho do conjunto de dados, do número de replicações e da variação média considerada. Neste projeto foram realizadas 10 replicações para obtenção dos resultados.

3.5.3 TEMPO DE SIMULAÇÃO

Cada execução da simulação teve a duração de 60 segundos. A capacidade de processamento do computador utilizado, e o tamanho do arquivo *trace* gerado pela simulação foram determinantes na estipulação do tempo, que se mostrou suficiente para obter os resultados desejados. O tempo de simulação foi definido pelo seguinte comando:

```
#Definindo Tempo de Simulação
set timesimu 60.0
```

A simulação com um maior tempo de duração gerava grandes registros de dados e o tratamento dos mesmos era impossibilitado pela pequena capacidade de processamento do computador utilizado.

O tempo de simulação em relação ao modelo adotado, a complexidade e a disposição dos nós na rede, implica na não necessidade de se manter um nível de precisão de tempo complexo, quando o interesse é estudar apenas uma determinada região do campo.

O capítulo seguinte apresenta a implementação das simulações realizadas, descrevendo o cenário adotado, as topologias e os scripts referentes à programação de cada simulação.

CAPÍTULO 4. SIMULAÇÕES DAS REDES IP CONVENCIONAL E MPLS

Neste capítulo será realizada a descrição das etapas e execução dos procedimentos, as topologias de rede utilizadas e a programação das simulações.

4.1 DEFINIÇÃO DA TOPOLOGIA DA REDE

A topologia de rede utilizada para a criação da simulação foi baseada em uma simplificação de um conjunto de roteadores, conhecido como *backbone*, projetado para atender requisitos técnicos e garantir largura de banda necessária ao tráfego Internet de produção (navegação Web, correio eletrônico, transferência de arquivos); ao uso de serviços e aplicações avançadas. [TANENBAUM, 2003].

O *backbone* da Rede Nacional de Pesquisa (RNP2), utilizado como referência neste projeto, possui pontos d epresença nos estados de São Paulo (SP), Rio de Janeiro (RJ), Distrito Federal (DF) e Minas Gerais (MG), sendo cada ponto de presença (PoP) dos estados, representado por nós da rede MPLS e da rede IP convencional. Nele os pacotes de dados são roteados com base nas informações contidas em seus cabeçalhos (*headers*) e nas informações que cada roteador dispõe sobre o alcance e a disponibilidade dos outros roteadores da rede.

Essa topologia foi escolhida por se tratar de uma topologia real além de ter seus pontos de presença distribuídos de tal forma que viabilizasse o uso da tecnologia em estudo. Foi alterada pela inclusão de mais um PoP (UNICEUB) e outros nós para a geração de tráfego background, com o objetivo de se criar uma rota alternativa diferente. A topologia está ilustrada na figura 4.1.

A Rede Nacional de Pesquisa (RNP) opera desde 1991 um backbone Internet que tem se modernizado de acordo com a demanda de seus usuários e, em particular, da comunidade acadêmica. Estando atualmente em sua fase II, o backbone da RNP possui como características a abrangência nacional e a velocidade adequada à utilização de aplicações do tipo Web. [SOUZA, 2002].

Além disso, essa rede é prática e muito utilizada em pesquisas científicas e de cunho acadêmico.

4.2 REDE IP - CONVENCIONAL

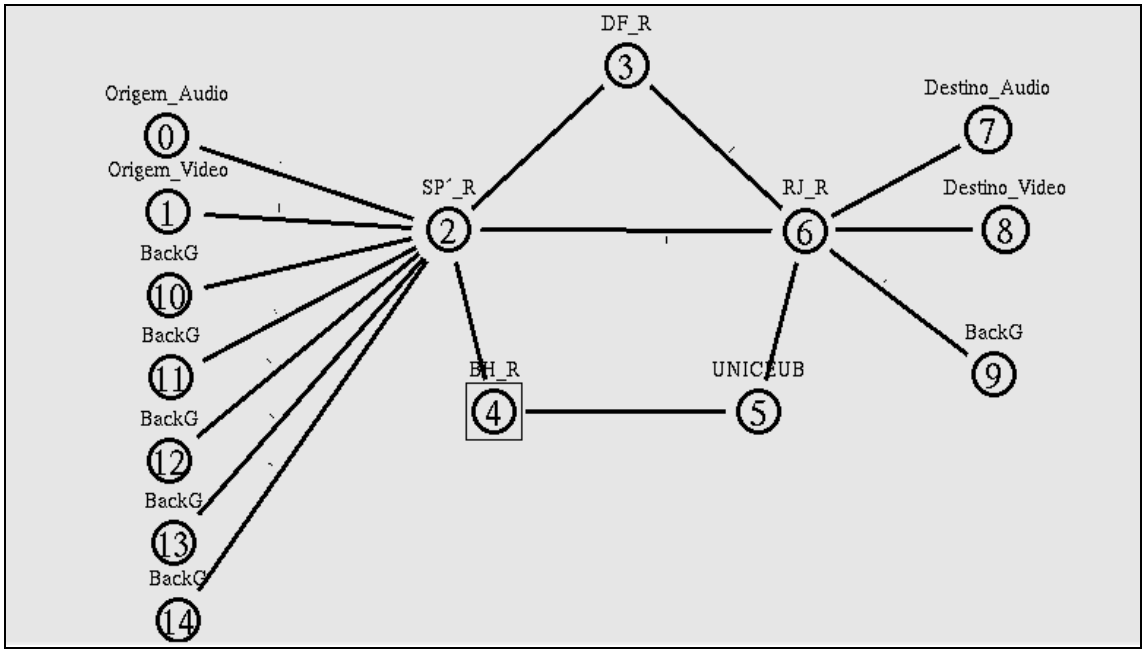

Figura 4. 1 - Topologia Rede IP Convencional

A rede IP teve sua arquitetura baseada na topologia da RNP representando neste trabalho uma rede IP convencional, como a Internet, por exemplo.

Esta rede IP é formada por 15 nós (roteadores IPs) numerados de 0 a 14, conforme mostra a figura 4.1.

O tráfego background, definido na figura como BackG, representa o tráfego de saturação utilizado para provocar congestionamento nos canais de comunicação.

Os nós BH_R, SP_R, RJ_R ,DF_R e o nó UNICEUB, representam os roteadores de núcleo e para representar o fluxo de audio e vídeo foram criados os nós Destino_Áudio/Destino_ Video e Origem_ Áudio/Origem_Vídeo.

A definição destes nós é ilustrada por um trecho do script *Otcl* [COUTINHO, 2003], apresentado na figura 4.2.

```
# Criando os Nós
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]
set n11 [$ns node]
set n12 [$ns node]
set n13 [$ns node]
set n14 [$ns node]
```

Figura 4. 2 - Criação dos nós da rede

A tabela 2.2 mostra uma descrição da rede IP com os nós, largura de banda e atrasos definidos para cada enlace. Os enlaces são do tipo duplex (comunicação feita entre um Transmissor e um Receptor, sendo que os dois podem transmitir dados simultaneamente em ambos os sentidos - transmissão é bidirecional) e utilizam a política de fila DropTail, que representa um algoritmo simples de controle de congestionamento no qual armazena os pacotes na ordem em que eles chegam, e, assim que a rede permitir, envia-os nesta mesma ordem.

Tabela 2. 2 - Descrição da Rede IP

Nó_Origem	Nó_Destino	Largura_Banda (Mb)	Atraso (ms)
0	2	12	8
1	2	10	8
2	6	12	8
2	3	12	8
2	4	10	8
4	5	20	5
3	6	18	5
5	6	16	10
6	7	10	20
6	8	12	8
6	9	10	8
10	2	10	8
11	2	10	8
12	2	10	8
13	2	10	8

O valores definidos acima são variáveis de testes, as quais possuem seus limites baseados na norma ITU-T a fim de representar situações mensuráveis próximas de aplicações reais durante a execução da simulação, visando um maior sucesso na obtenção dos resultados.

O trecho do script de simulação que define o tipo enlace, a largura de banda, o atraso e a política de fila é mostrado na figura 4.3:

```
# Criando os Links
$ns duplex-link $n0 $n2 10Mb 8ms DropTail
$ns duplex-link $n1 $n2 10Mb 8ms DropTail
$ns duplex-link $n2 $n6 12Mb 8ms DropTail
$ns duplex-link $n2 $n3 12Mb 8ms DropTail
$ns duplex-link $n2 $n4 10Mb 8ms DropTail
$ns duplex-link $n4 $n5 20Mb 5ms DropTail
$ns duplex-link $n3 $n6 18Mb 5ms DropTail
$ns duplex-link $n5 $n6 16Mb 10ms DropTail
$ns duplex-link $n6 $n7 10Mb 20ms DropTail
$ns duplex-link $n6 $n8 12Mb 8ms DropTail
$ns duplex-link $n6 $n9 10Mb 8ms DropTail
$ns duplex-link $n10 $n2 10Mb 8ms DropTail
$ns duplex-link $n11 $n2 10Mb 8ms DropTail
```

```
$ns duplex-link $n12 $n2 10Mb 8ms DropTail
$ns duplex-link $n13 $n2 10Mb 8ms DropTail
$ns duplex-link $n14 $n2 10Mb 8ms DropTail
```

Figura 4. 3 - Criação dos enlaces

4.3 REDE MPLS

A arquitetura de rede utilizada para introduzir um domínio MPLS na rede IP é formada pelos nós, agora conhecidos como LSRs: 2 (SP), 3 (DF), 4 (BH), 5 (UNICEUB) e 6 (RJ).

Os demais nós permaneceram representados como roteadores IPs. O nó 2 está disposto no domínio MPLS como um LSR ingresso e o nó 6 como um LSR egresso. Já os outros pontos de presença representam os LSRs de núcleo.

A figura 4.4 ilustra a rede simulada com MPLS.

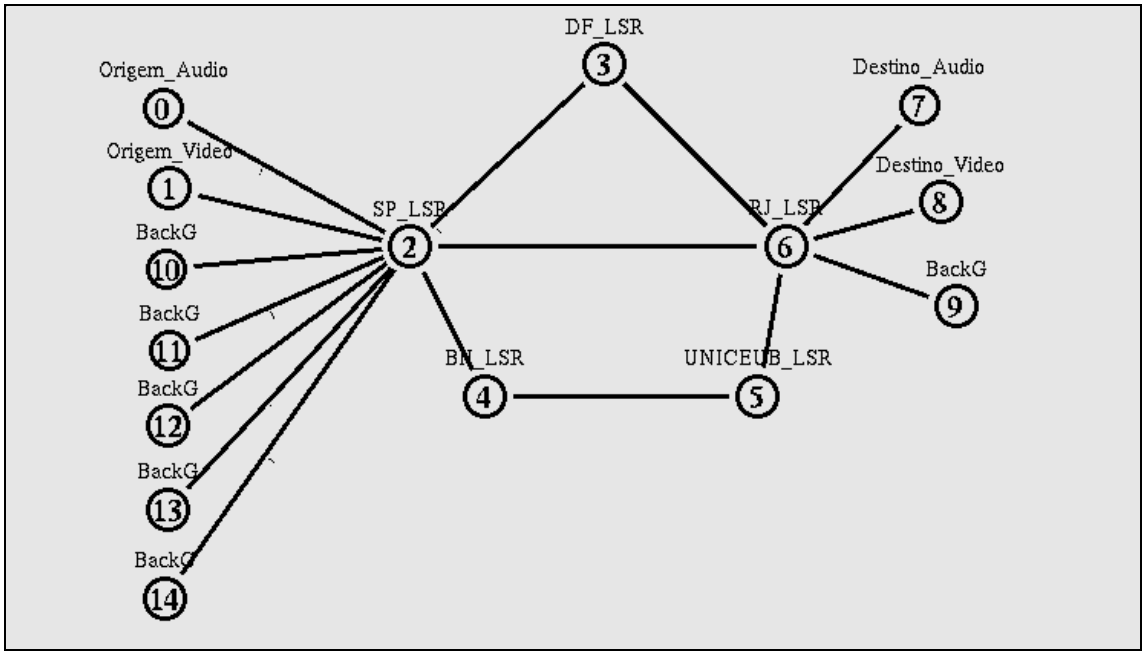

Figura 4. 4 - Topologia MPLS

As políticas de fila, os enlaces e os atrasos permanecem os mesmos da rede IP convencional, sendo alterada apenas a definição dos nós para representar a criação de um domínio MPLS dentro da rede IP.

O trecho do código que define o domínio MPLS é mostrado na figura 4.5.

```
# Criando os Nos
set n0 [$ns node]
set n1 [$ns node]
$ns node-config -MPLS ON
set LSR2 [$ns node]
set LSR3 [$ns node]
set LSR4 [$ns node]
set LSR5 [$ns node]
set LSR6 [$ns node]
$ns node-config -MPLS OFF
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]
set n11 [$ns node]
set n12 [$ns node]
set n13 [$ns node]
set n14 [$ns node]
```

Figura 4. 5 - Criação dos nós da rede MPLS

4.4 TRÁFEGO DA REDE

Na rede IP convencional e na rede MPLS foram inseridas aplicações geradoras de fontes híbridas de tráfego (áudio, vídeo e dados). Foram criadas, para isto, algumas conexões. Sendo 9 (nove) TCPs, utilizadas no tráfego de dados (background) e 2 (duas) UDPs para o tráfego de áudio e vídeo.

O tráfego de áudio e vídeo é representado por aplicações CBRs (*Constant Bit Rate*) e o tráfego de dados por aplicações FTPs.

As aplicações CBRs consiste em um termo relacionado com a Qualidade de Serviço (QoS), onde o principal objetivo dessa classe, é suportar aplicações de tempo real tais como vídeo ou streaming de voz. Normalmente o fluxo CBR possui uma maior prioridade quando comparado à outros fluxos. [ARC CHART, 2004].

Os fluxos relevantes para esta análise são os que representam maior complexidade, como o de áudio e vídeo pois essas aplicações foram inseridas em conexões UDPs (não confiáveis) para simular o serviço de melhor esforço, oferecido pela rede IP convencional.

A conexão UDP para o tráfego de áudio foi criada entre os nós 0 e 7. Considerando uma taxa de bits constantes (CBR) com pacotes de tamanho igual a 375 bytes enviados com intervalo de transmissão de 3ms.

Já para o tráfego de vídeo foi criada outra conexão UDP entre os nós 1 e 8. Nesta conexão foram inseridos pacotes do mesmo tamanho e intervalo de transmissão a uma taxa de aproximadamente 1000 kbps.

A programação que define as conexões UDPs e o tráfego de áudio e vídeo é apresentada na figura 4.6:

```
# Criando duas conexões UDPs
# Primeira conexão UDP1 entre os nós 0 e 7
set udp1 [new Agent/UDP]
set null1 [new Agent/Null]
$ns attach-agent $n0 $udp1
$ns attach-agent $n7 $null1
$ns connect $udp1 $null1
$udp1 set class_ 3
# Segunda conexão UDP1 entre os nós 1 e 8
set udp1 [new Agent/UDP]
set null1 [new Agent/Null]
$ns attach-agent $n1 $udp1
$ns attach-agent $n8 $null1
$ns connect $udp1 $null1
$udp1 set class_ 3
```

Figura 4. 6 - Criação dos nós da rede

Foram divididas as conexões TCPs para o tráfego de dados background (FTP) entre os nós 10, 11, 12, 13 e 14 com destinos direcionados aos nós 6 e 9. Para os nós 11, 12 ,13 e 14 foram definidas oito conexões (duas para cada nó) a fim de maximizar o fluxo de dados e conseqüentemente o congestionamento na rede. Todas as conexões TCPs foram configuradas para

transmitir pacotes com tamanho de 1000 bytes representando o tráfego de melhor esforço.

4.5 DEFINIÇÃO DO TEMPO DE SIMULAÇÃO

Cada execução da simulação teve a duração de 60 segundos. Para as fontes CBR e FTP, foi definido o inicio da transmissão no instante 0 (zero) de tempo.

4.6 EXECUÇÃO DA SIMULAÇÃO

As simulações foram modeladas com o intuito de se criar um panorama de congestionamento na rede.

No primeiro momento foi simulada uma rede IP tradicional formada por roteadores IPs e utilizando-se do processo de encaminhamento padrão da Internet (melhor esforço).

A tabela 2.3 ilustra a programação da simulação com os tempos de início e fim e os instantes que iniciaram e finalizaram os tráfegos gerados.

Tabela 2. 3 - Simulaçao da Rede IP

Instante_Simulação	Evento
0.00	**Inicia simulação**
0.05	**Inicia tráfego de áudio e vídeo (CBR)**
0.10	**Inicia tráfego FTP de dados (Backgraund) [1º]**
0.20	**Inicia tráfego FTP de dados (Backgraund) [2º]**
60	**Finaliza tráfego de áudio e vídeo**
60	**Finaliza tráfegos de dados**
60	**Finaliza simulação**

Para visualização no nam, os pacotes de áudio e vídeo foram representados pelas cores vermelha e verde respectivamente e os pacotes de dados foram representados pela cor azul.

A figura 4.7 mostra o instante em que os pacotes de áudio e vídeo começam a se enfileirar (verticalmente para cima) juntamente com os pacotes de dados. Observe que o pacotes seguem o menor caminho, ou seja rota (enlace) que liga os nós 2 e 6 definido pelo serviço de melhor esforço. Os

pacotes de áudio seguem o fluxo nos nós 0 e 7 e os pacotes de vídeo seguem pelos nós 1 e 8.

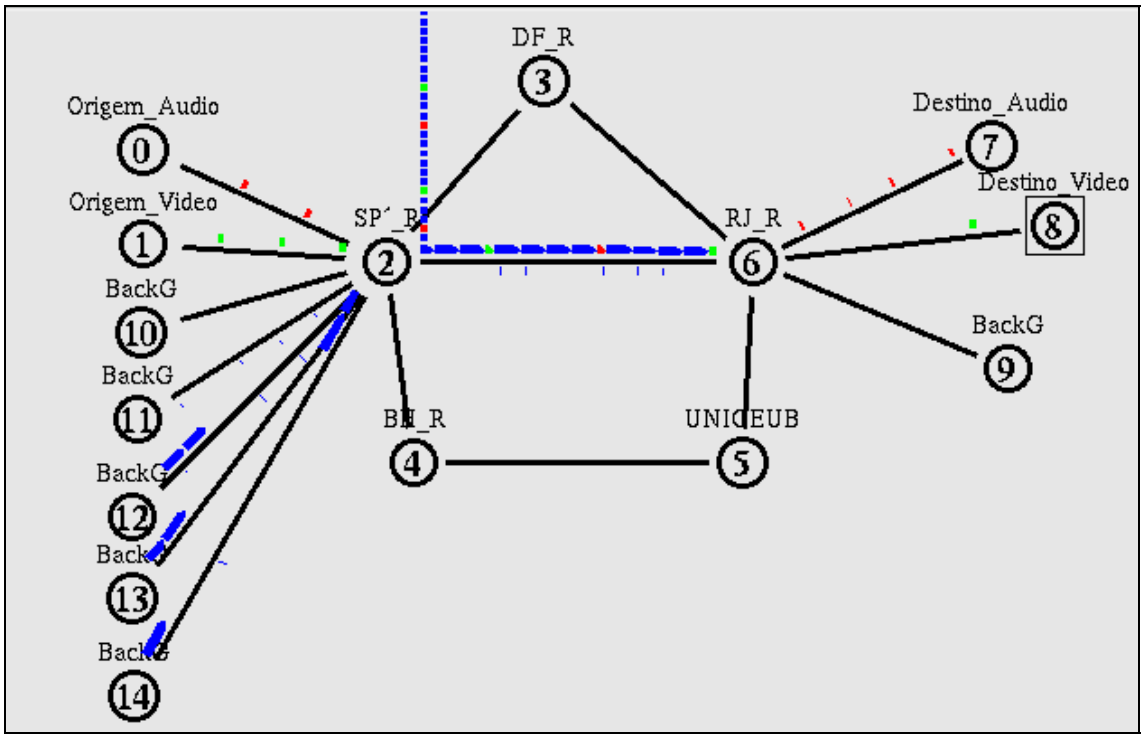

Figura 4. 7 - Enfileiramento de pacotes

A figura 4.8 apresenta o descarte de pacotes (verticalmente para baixo) de áudio e vídeo em consequência do congestionamento criado entre os nós 2 e 6, representado pelo menor caminho.

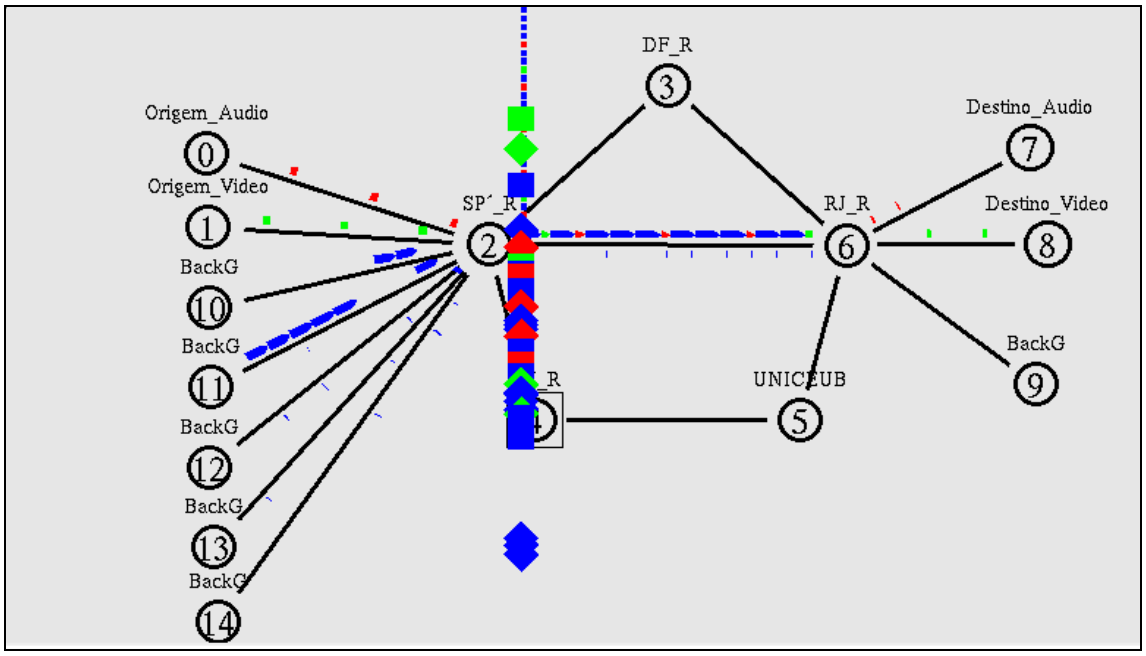

Figura 4. 8 - Perda de pacotes

Em seguida, foi simulada a rede MPLS a partir da definição de LSPs para o tráfego de áudio e vídeo.

A tabela 2.4 mostra os tempos de início e fim da simulação e dos tráfegos gerados.

Tabela 2. 4 - Simulaçao da Rede MPLS

Instante_Simulação	Evento
0.00	Inicia simulação
0.04	Inicia tráfego de Áudio (CBR)
0.05	Inicia tráfego de Vídeo (CBR)
0.10	Inicia trafego FTP de dados (Backgraund) [1º]
0.20	Inicia trafego FTP de dados (Backgraund) [2º]
0.15	Envio de LDP para E-LSR
0.1	Envio de mensagem para LSR Ingresso (áudio e vídeo)
0.25	Criação de LSP para áudio e vídeo
60.00	Finaliza tráfego de áudio
60.00	Finaliza tráfego de vídeo
60.00	Finaliza tráfegos de dados
60.00	Finaliza roteamento para áudio e vídeo
60.00	Finaliza simulação

A figura 4.9 apresenta a criação do domínio MPLS a partir da inserção dos LSPs para o tráfego de áudio e vídeo. Neste caso, os pacotes de áudio (vermelhos) trafegam agora pelo nó 3 e os pacotes de vídeo (verdes) pelos nós 4 e 5.

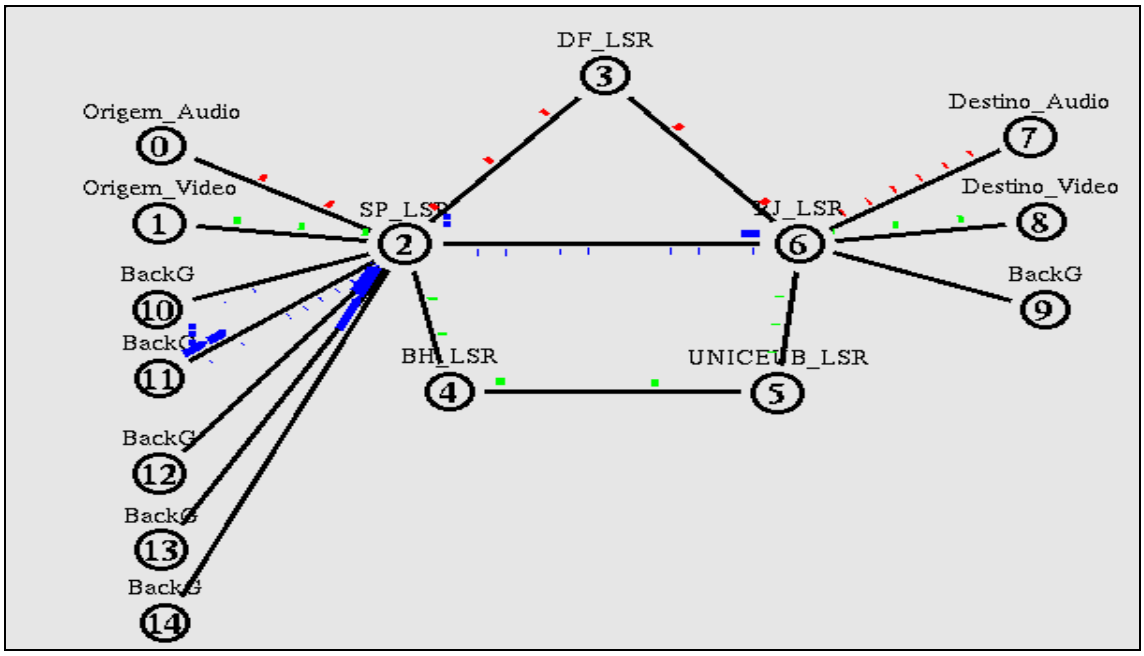

Figura 4. 9 - LSP's

Nessa simulação foi utilizado o simulador NS, na versão ns-2.29 e as simulações foram executadas em uma máquina com processador Celeron de 2.6 GHz, 256 Mb de memória RAM, executando o sistema operacional Linux Kurumim.

Foram realizados 10 replicações de simulações de 60 segundos para gerar o histórico de análise.

No próximo capítulo serão apresentados os resultados obtidos nas simulações de rede aqui descritas.

CAPÍTULO 5. SIMULAÇÕES E RESULTADOS

Este capítulo apresenta a análise e avaliação do comportamento das aplicações de dados, áudio e vídeo através de gráficos gerados a partir do histórico coletado na simulação. Essas aplicações foram contempladas em dois cenários diferentes. Um cenário representado pela rede IP convencional que simula serviço de melhor esforço, utilizado pelas redes IPs, e outro representado por uma simulação de rede no domínio MPLS.

Considerando a análise dos tráfegos prioritários de áudio e vídeo, foram definidos como parâmetros de QoS: o atraso, a vazão e o descarte de pacotes. A análise realiza a comparação dos parâmetros de qualidade de serviço alcançados pela rede IP e os resultados obtidos através da implementação da tecnologia MPLS.

5.1 FLUXO DE ÁUDIO

O fluxo de áudio é um dos tráfegos representados por aplicações CBRs, que necessita de um tratamento específico. Essas aplicações foram inseridas em conexões não confiáveis (UDPs) para retratarem o atual serviço oferecido pela Internet.

Neste caso, é enfatizado a otimização da rede, com objetivos relacionados a QoS como menor atraso, alta taxa de transmissão, pequena perda de pacotes e serviço previsível.

O tráfego foi criado entre os nós 0 e 7, conforme ilustrado na figura 5.1
O tamanho de pacotes é de 375 bytes enviados com intervalo de transmissão de 3ms. Foi considerado, na escolha desses atributos, o cenário utilizado, o meio físico, e a largura de banda adotada na simulação.

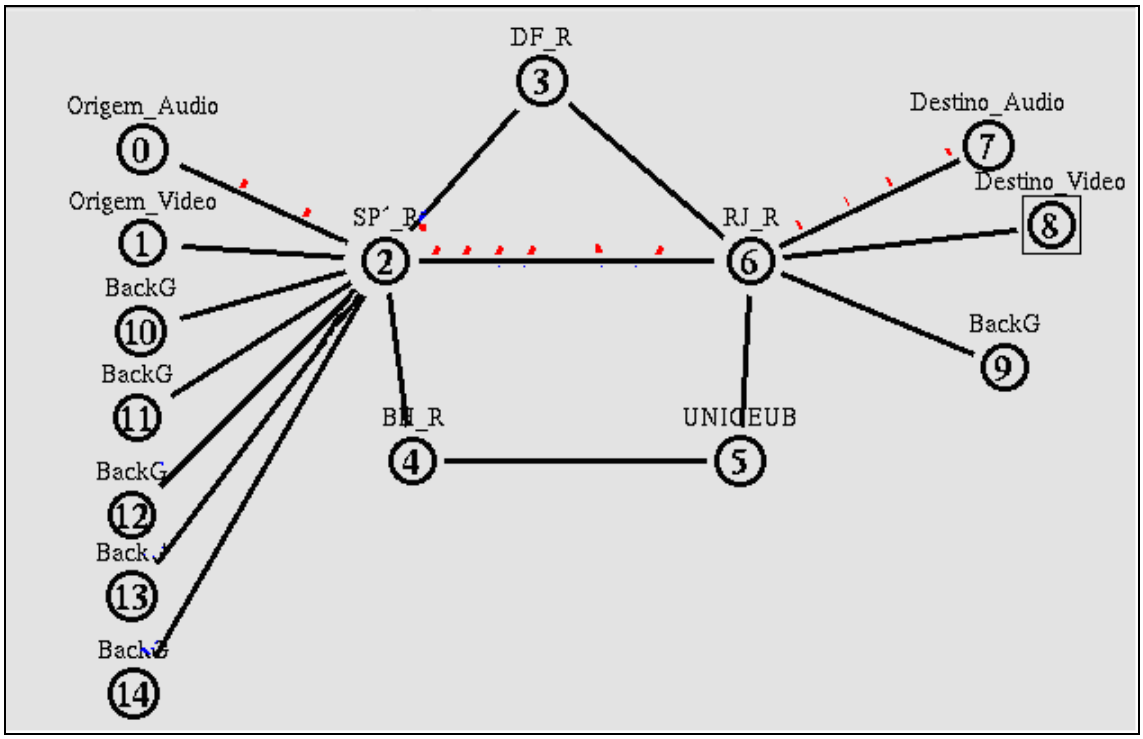

Figura 5. 1 - Tráfego nos nós 0 e 7

5.1.1 Atraso

O fluxo de áudio com o serviço IP de melhor esforço sofreu atrasos em vários períodos da simulação, variando entre 40ms e 52ms aproximadamente de atraso em determinados momentos. Contudo, os atrasos não ultrapassaram o limite máximo de 150ms definidos pela ITU –T. [M.PERKINS]

Já com a implementação do MPLS, houve uma redução considerável no atraso dos pacotes do fluxo de áudio. Como pode ser observado na figura 5.2, houve estabilidade dos atrasos em aproximadamente 40ms durante o processo de simulação, devido a priorização do tráfego gerado nos nós Origem_Àudio e Destino_Áudio.

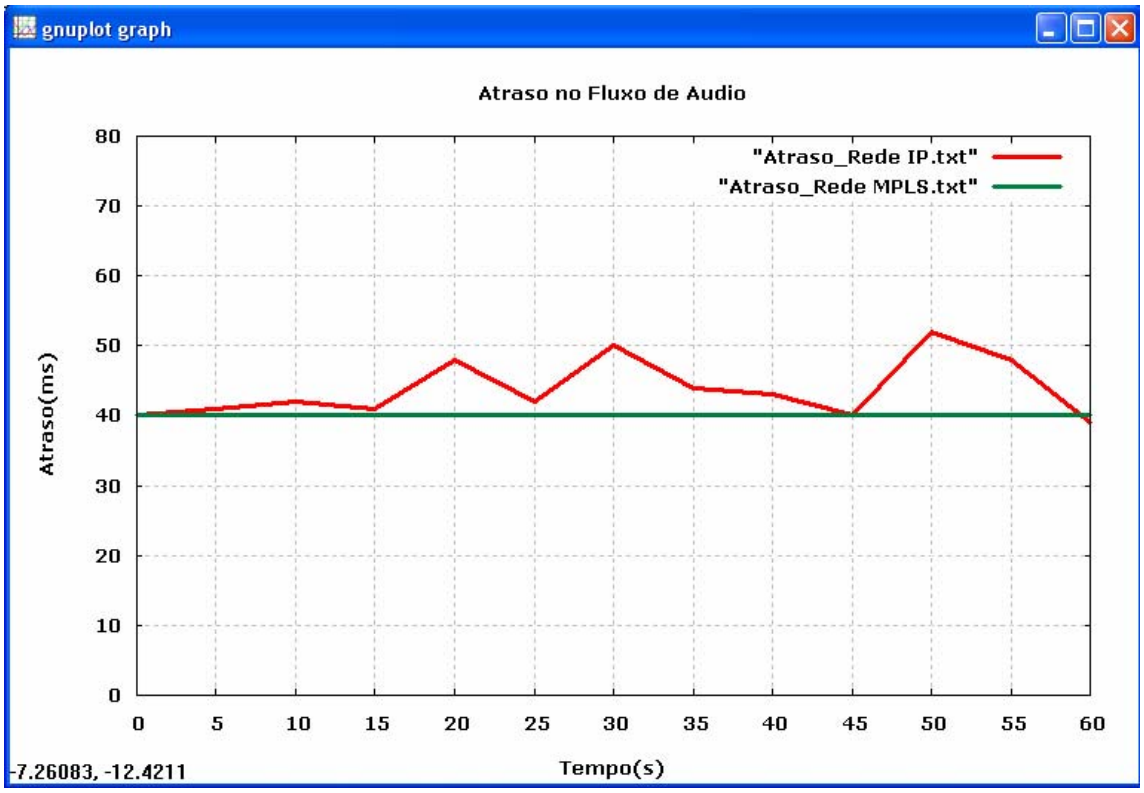

Figura 5. 2 - Atraso no Fluxo de Áudio

Sendo assim, a adição do serviço MPLS, atendeu eficientemente o limite estabelecido pela ITU –T, de 150ms.

Embora o atraso seja semelhante em ambas as redes, com o MPLS a comportamento estável favorece o restauro rápido de rotas possibilitando economias de escala (escalabilidade). [ROSEN]

5.1.2 Vazão

O tráfego de áudio sem MPLS atingiu picos de vazão durante a simulação, apresentando uma variação entre 0.095Mbits/s e 0.1Mbits/s, aproximadamente.

Aplicando MPLS, a vazão de áudio obtida foi de aproximadamente 0.3Mbits/s permanecendo quase estável até o momento de finalização da simulação, conforme ilustrado na figura 5.3. Segundo os parâmetros da norma ITU-T, o mínimo exigido é de 0.1 Mbps.

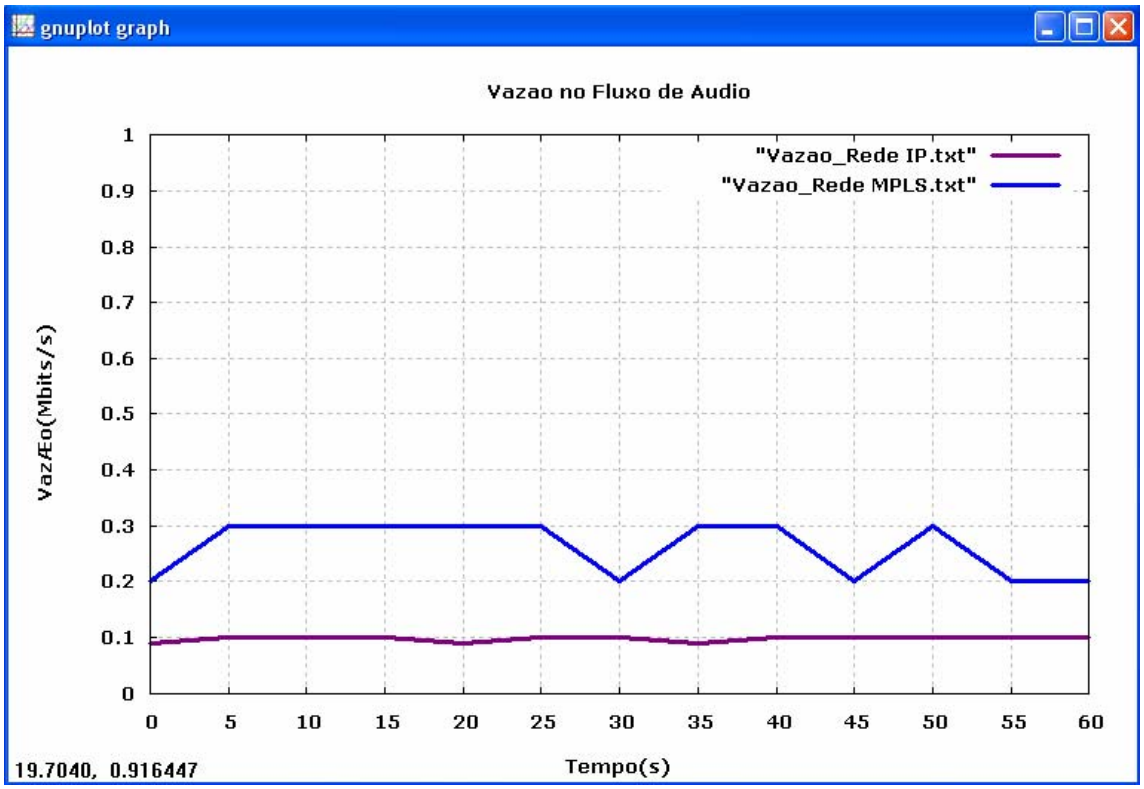

Figura 5. 3 - Vazão no Fluxo de Áudio

5.1.3 Descarte de pacotes

O descarte de pacotes na rede IP ultrapassou o limite máximo estabelecido, nos intervalos 30s a 40s atingindo cerca de 2% de descarte de pacotes.

Segundo a ITU-T, é permitido como limite estabelecido para a transmissão de áudio dos pacotes, um descarte de no máximo 1%. Logo, essa recomendação não foi atendida a contento.

Pode-se observar na figura 5.4, que com a utilização da tecnologia MPLS, o tráfego de áudio flui de maneira eficiente. Durante a simulação não houve perda de pacotes significativa, atingindo aproximadamente de 0.9% a 1%, o que viabiliza a transmissão de áudio.

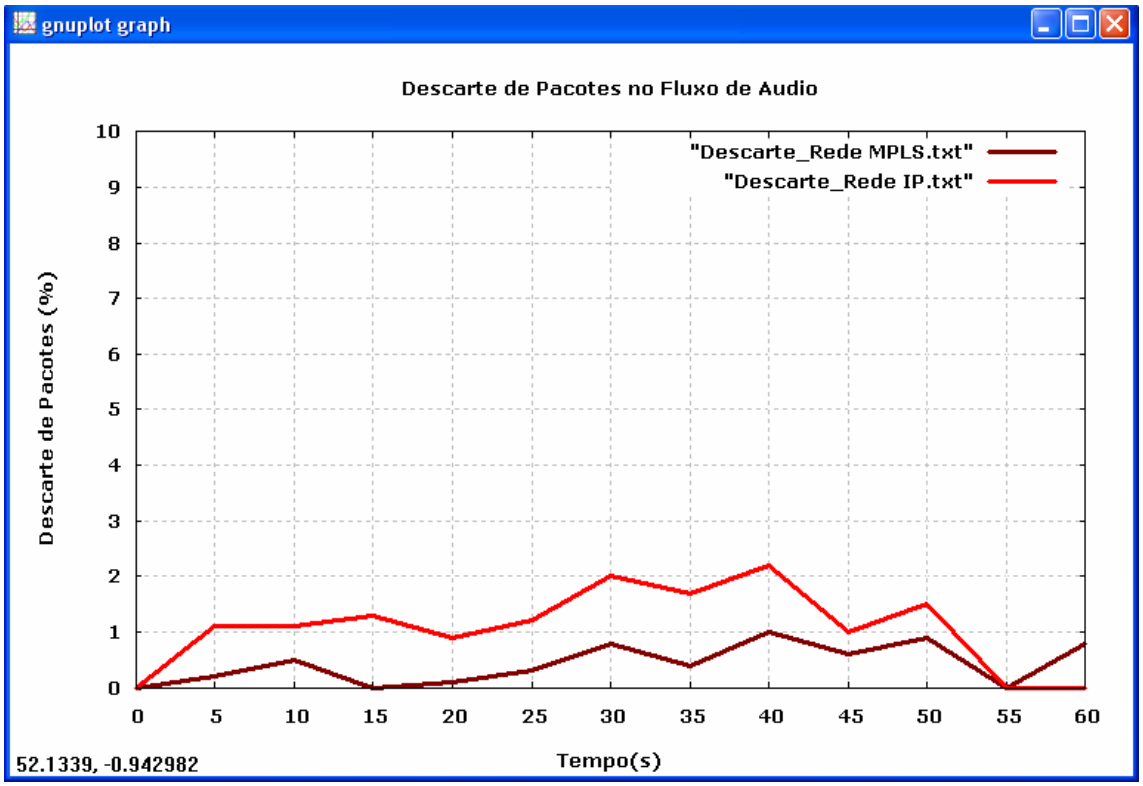

Figura 5. 4 - Descarte no Fluxo de Áudio

A implementação do tráfego através dos LSPs permitiu que a transmissão dos pacotes ocorresse sem perdas significativas, ou seja, após a criação dos LSPs os pacotes de aplicações complexas e sensíveis (áudio e vídeo) não sofreram descartes expressivo, tampouco enfileiramento na transmissão.

5.2 FLUXO DE VÍDEO

O fluxo de vídeo, também representa um tráfego utilizado em aplicações CBRs, inseridas em conexões UDPs que buscam otimizar a rede simulada, de forma que atenda todos os parãmetros de QoS aplicados, como, menor atraso, alta taxa de transmissão, pequena perda de pacotes e serviço previsível.

O tráfego foi criado entre os nós 1 e 8, como tamanho de pacotes e intervalo de transmissão semelhante ao adotado no fluxo de áudio, conforme figura 5.5.

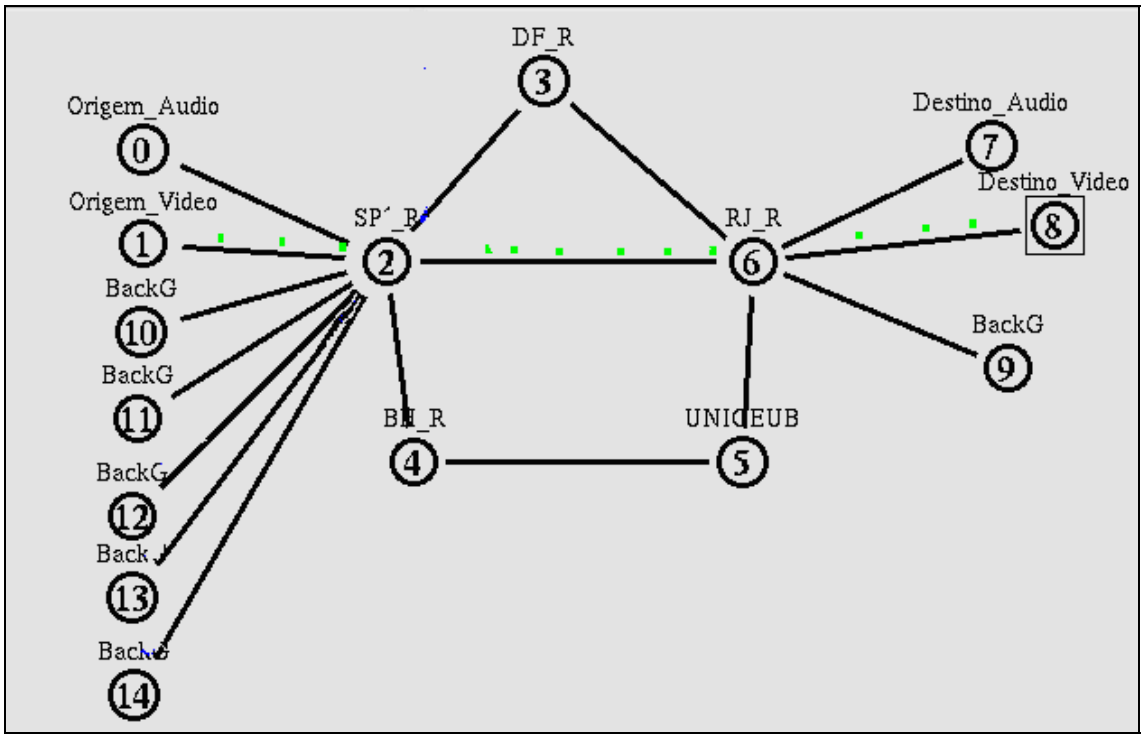

Figura 5. 5 - Tráfego nos nós 1 e 8

5.2.1 Atraso

O tráfego de vídeo em redes IP sofreu atrasos significativos durante a simulação, chegando a atingir cerca de 55ms em alguns instantes específicos da simulação.

A aplicação do MPLS nesse fluxo possibilitou o tráfego de pacotes de vídeo com um atraso em torno de 38 a 39ms ao longo da simulação, o que atende o limite máximo de 150ms definido pela ITU –T. conforme ilustra a figura 5.6.

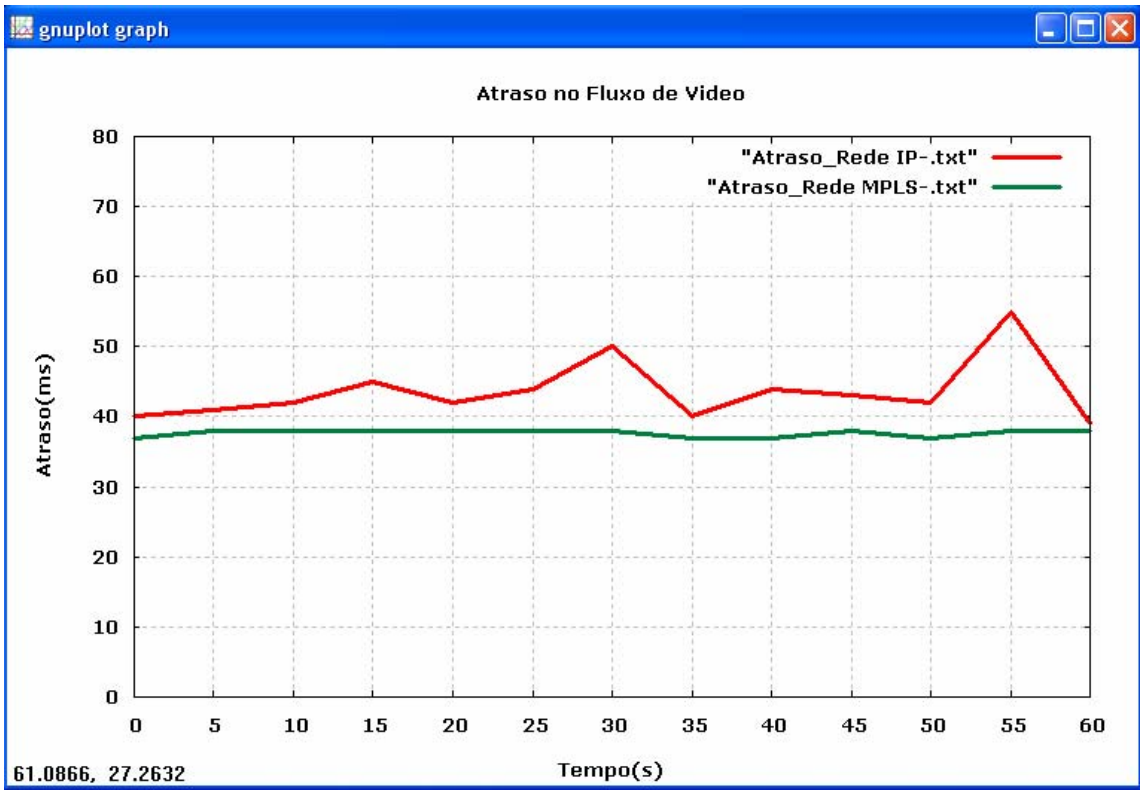

Figura 5. 6 - Atraso no Fluxo de Vídeo

5.2.2 Vazão

Em virtude do congestionamento, o gráfico de vazão apresentou variação aproximada de 0.9 a 1Mbits/s , conforme ilustração da figura 5.7.

O tráfego de vídeo na rede MPLS atingiu uma vazão de aproximadamente 1.3Mbits/s, viabilizando o tráfego de vídeo e atingindo o mínimo definido pela ITU –T, para essas aplicações.

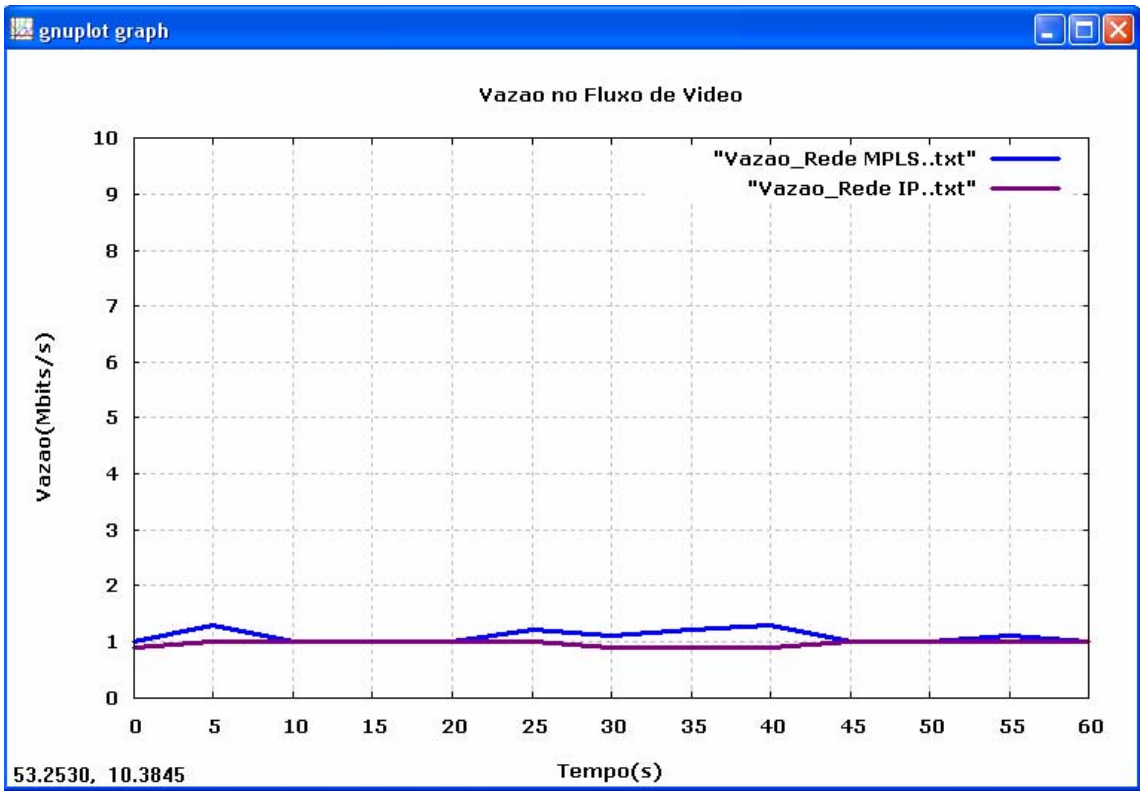

Figura 5. 7 - Vazão no Fluxo de Vídeo

-

5.2.3 Descarte de pacotes

A perda de pacotes no fluxo de vídeo, representado na figura 5.8, mostra que nos intervalos de 15s a 25s e 35s a 50s, houve descarte excessivo dos pacotes de vídeo devido ao expressivo congestionamento formado entre os nós 2 e 6.

Através da implementação do serviço de MPLS nota-se a que a perda de pacotes de vídeo foi em torno de 1%, porém com conseqüências positivas, por se tratar de uma transmissão sensível, além de atender às recomendações e limites regulamentados pela Organização dos Padrões de Telecomunicações (ITU-T).

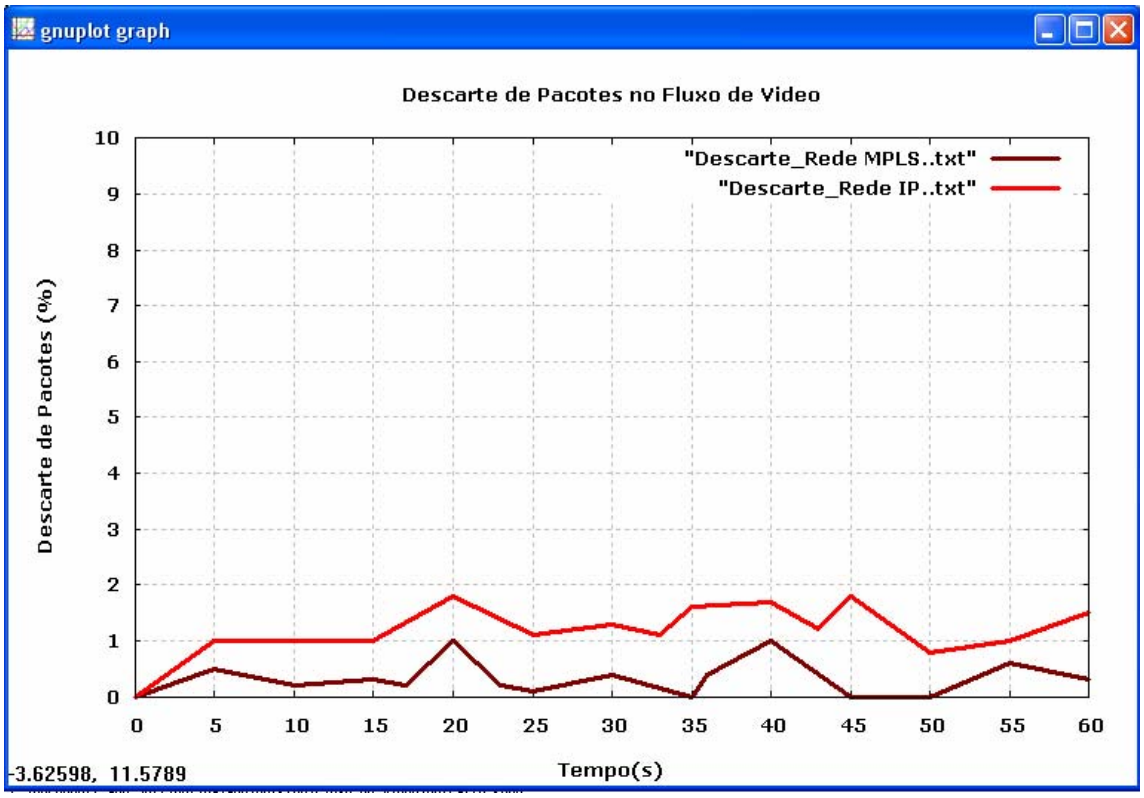

Figura 5. 8 - Descarte no Fluxo de Vídeo

Em linhas gerais, observou-se com a análise aos fluxos prioritários, áudio e vídeo, que a rede IP através de seu serviço (best-effort) não oferece condições satisfatórias para que essas aplicações sejam realizadas. Tal afirmação está baseada nos parâmetros exigidos pelo regulamento que rege a qualidade de serviço no âmbito das telecomunicações.

Ao ser introduzido um domínio MPLS nessa rede e aplicado tratamento diferencial nesses fluxos, constatou-se a possibilidade de obter níveis de QoS que atendesse serviços seguros, previsíveis e mensuráveis.

Analisando a aplicação de vídeo, notou-se que o serviço de melhor esforço não conseguiu prover os parâmetros de QoS com eficiência. Utilizando a tecnologia MPLS, essa aplicação passou a representar um serviço mais confiável que proporcionou o alcance dos padrões estabelecidos pela norma referenciada.

CAPÍTULO 6. CONCLUSÃO

Compreender o funcionamento da Internet, prever o seu crescimento e propôr novos protocolos e mecanismos de provisão de QoS são atividades complexas, mas que podem ser obtidas com a realização de simulações.

Os resultados da simulação comprovam a possibilidade de obtenção de melhorias no desempenho e QoS em uma rede com a utilização da tecnologia MPLS.

Através das simulações ficou evidenciado que as exigências mínimas estabelecidas pela norma ITU-T, para provê melhor qualidade nas aplicações de voz e vídeo foram garantidas com o emprego do método de comutação de pacotes. O emprego do MPLS possibilita um aumento significativo da vazão obtida pelos fluxo de voz e vídeo e atenua expressivamente a perda de pacotes do fluxo.

Para serviço de melhor esforço na rede IP convencional, foi demonstrado que o mesmo não atendeu as exigências mínimas das aplicações de voz e vídeo, com relação aos parâmetros de vazão e perda de pacotes, avaliados na rede MPLS. Assim, ficou provada a possibilidade de se oferecer níveis de QoS aceitáveis e bem mais satisfatórios do que em redes IPs.

A tecnologia proposta neste projeto se mostrou de grande utilidade, no que tange ao provimento de QoS. Além de prover escalabilidade em determinados instantes da simulação, ela, por meio da agregação de fluxos ofereceu alcance dos requisitos de qualidade necessários estabelecidos pela norma ITU – T. para as aplicações de áudio e vídeo, através da engenharia de tráfego.

Essas exigências foram atendidas, evitando a necessidade de executar procedimentos de sinalização de pacotes em cada um dos roteadores pertencentes à rede, o que aumentava seu processamento.

Além disso, a tecnologia proposta neste projeto também é capaz de distinguir e tratar as diferentes classes de tráfego, que exigem características de serviço específicas e que devem ser garantidas por todo o caminho ao longo da rede.

Nesse cenário, o MPLS demonstra sua importância, em larga escala, quando direcionado à convergência de serviços simultâneos de áudio, vídeo e dados bem como outras aplicações.

Evidencia-se que, apesar de não existir ainda nenhuma solução global que atenda todos os requisitos de QoS, é possível melhorar o desempenho das redes de comunicação através de medidas menos complexas de QoS, como as utilizadas neste projeto de pesquisa.

6.1 Objetivos alcançados e Contribuições

Como conclusão final dos resultados obtidos, válida para as simulações realizadas e mostradas nos capítulos anteriores, pode-se afirmar que a tecnologia proposta foi satisfatória em seus resultados, quando testado no cenário configurado e com diferentes tipos de parâmetros que medem a qualidade de serviço de uma rede.

Os gráficos apresentados nos capítulos anteriores mostram que a visualização dos resultados melhora com o aumento das escalas utilizadas, melhorando notoriamente a precisão da leitura realizada na simulação.

A análise e comparação de resultados, embora fundamental do ponto de vista da análise de performance, não deveria esconder o fato de que sua abordagem apresenta não uma forma competitiva de alcançar valores mínimos de qualidade, mas sim um novo paradigma de roteamento.

As comparações dos resultados foram feitas com a intenção de mostrar que a tecnologia proposta é uma alternativa viável a ser utilizada no problema de congestionamento de tráfego. Os resultados numéricos das simulações confirmam este propósito, já que eles se apresentaram satisfatórios e aceitáveis pela norma ITU-T.

Isto dá lugar a afirmar que a tecnologia proposta é válida. Desta maneira, o objetivo principal do trabalho, foi alcançado de forma primordial.

O desenvolvimento deste projeto contribuiu de forma efetiva para o estudo de soluções de Engenharia de Tráfego para redes IP, impulsionando a popularização das redes locais com MPLS e colaborando fortemente para a escalabilidade e robustez das redes atuais.

6.2 Dificuldades Encontradas

As principais dificuldades encontradas ao longo do projeto foram principamente a definição de uma topologia para apresentar as características inerentes à realização das simulações, de forma que apresentasse um gargalo e que fosse baseada em uma rede já existente para tornar a simulação mais próxima da real e a instalação e configuração do Network Simulator (NS), que é feita de forma não automatizada e de grande complexidade, também exigiu muito esforço.

A escassez de material didático e literatura sobre simulação de redes e o simulador NS, a modelagem dos cenários de simulação, a programação das simulações e a análise do arquivo trace gerado pelas simulações devido à capacidade de processamento e memória do equipamento utilizado, também representaram um complicador no processo de formatação dos dados para análise.

6.3 Trabalhos Futuros

Como trabalhos futuros nessa área inclui-se a utilização de topologias mais complexas, a interação entre vários domínios integrados a Diffserv e a utilização de Engenharia de Tráfego para o fluxo destinado ao tráfego de melhor esforço.

Além disso, é importante pesquisar outras técnicas para evitar o desperdício de recursos quando não existem caminhos alternativos, que não utilizem Engenharia de Tráfego.

REFERÊNCIAS BIBLIOGRÁFICAS

[ASHWOOD] ASHWOOD-SMITH, P. & JAMOUSSI, B., “MPLS Tutorial”, 1999, http://www.nanog.org/mtg-9905/ppt/mpls/index.htm

[ARC CHART]Arc Chart. Wireless to the max – WiMAX. 19 June 2003. Disponível ehttp://www.arcchart.com/blueprint/show.asp?id=328&qtabs=99999 [Acesso em:Março de 2007]

[COUTINHO] COUTINHO, Mauro Margalho. *Network Simulator – Guia Básico para Iniciantes.* UFPa, 2003.

[FAUCHER] FAUCHER, F. L. et. al., “MPLS Support of Differentiated Services”, Internet Draft, <draft-ietf- MPLS-diff-ext-03.txt>, Fevereiro 2000.

[GALO] GALO, Mauricio. Instruções iniciais para uso do GNUPLOT. Disponível em:<http://www.prudente.unesp.br/dcartog/galo/gnuplot>. Acesso em: 09 abril 2007.

[GAY] GAY, Warren W. Programação para Linux. São Paulo: Campus, 1999. ISBN 85-352-0474-1.

[GRANADO] GRANADO, Filho, Arlindo Garcia, Trabalho sobre MPLS, Faculdade de Elétrica e de Computação – Universidade Estadual de Campinas, dezembro 1998. Disponível em <www.cisco.com.br > Acesso em 13/04/2007 às 02:10h.

[MURHAMMER] MURHAMMER, Martin W.; ATAKAN, Orcun; BRETZ, Stefan; PUGH, Larry R.; SUZUKI, Kazunari; WOOD, David H.; TCP/IP Tutorial e Técnico. São Paulo, 2000: Makron Books do Brasil Editora Ltda.[CISCO] www.cisco.com.br - “Software Cisco Provisioned QoS Policy Manager 2.0”.

[M.PERKINS] M.Perkins, Thorpe. "Characterizing the subjective performance of the ITU-T 8 kb/s speech coding algorithm-ITU-T ". IEEE Communications Magazine, pp. 74-81, september, 1997.

[OLIVEIRA] OLIVEIRA, Renato D. V. Msc.; Dias, Roberto A. *Multiprotocol Label Switching.* Disponível em www.ucer.nurcadufsc.br/docs/mpls2.ppt, acessado em 23/03/07 às 10:30:00h.

[ROSEN] ROSEN, E. *Multiprotocol Label Switching Archtecture,* Request For Comments 2212, Setembro de 1997. http://www.ietf.org/rfc2212.txt

[SILVA] SILVA, Dinailton José da. Análise de Qualidade de Serviço em Redes. 2004.(Dissertação de Mestrado em Ciência da Computação) – Universidade Estadual de Campinas.

[SOARES] SOARES, Luiz Fernando G; LEMOS, Guido; COLHER, Sérgio. Redes de computadores: das LANs, MANs e WANs às redes ATM. 2. ed. Rio de Janeiro: Campus, 1995.

[SOUZA[SOUZA, FASOLO. Desempenho da Modulações QPSK e pi/4-DQPSK em um canal kappa-mu com Desvanecimento Lento Utilizando Diversidade MRC In: IV Workshop de Comunicações Sem Fio e Computação Móvel, 2002, São Paulo

[TANENBAUM] TANENBAUM, Andrew S. *Redes de computadores.* 6. ed. Rio de Janeiro: Editora Campus, 2003.

ANEXO A

```
# Autor: Daiane Vaz Lima
# Data: Março de 2007.
# Nota: Rede IP Convencional
set ns [new Simulator]
set cir0  1000000
set rate0 8000000
set testTime 60.0
set packetSize 1000
# Ajustando cores para animação no nam
#Cor do pacote CBR (voz)
$ns color 1 red
#Cor do pacote CBR (video)
$ns color 3 green
#Cor dos pacotes FTPs
$ns color 2 blue
# Arquivo de trace
set ni [open redeip.nam w]
$ns namtrace-all $ni
# Abrindo parâmetros do arquivo de trace
set nil [open logip.tr w]
$ns trace-all $nil
#
# Definindo a Rede
# Criando os Nós
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
```

```
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]
set n11 [$ns node]
set n12 [$ns node]
set n13 [$ns node]
set n14 [$ns node]
# Criando os rótulos para os Nós
$n0 label "Origem_Áudio"
$n1 label "Origem_Video"
$n2 label "SP´_R"
$n3 label "DF_R"
$n4 label "BH_R"
$n5 label "UNICEUB_R"
$n6 label "RJ_R"
$n7 label "Destino_Audio"
$n8 label "Destino_Video"
$n9 label "BackG"
$n10 label "BackG"
$n11 label "BackG"
$n12 label "BackG"
$n13 label "BackG"
$n14 label "BackG"
# Criando os Links
$ns duplex-link $n0 $n2 10Mb 8ms DropTail
$ns duplex-link $n2 $n6 12Mb 8ms DropTail
$ns duplex-link $n2 $n3 12Mb 8ms DropTail
$ns duplex-link $n2 $n4 10Mb 8ms DropTail
$ns duplex-link $n4 $n5 20Mb 5ms DropTail
$ns duplex-link $n3 $n6 18Mb 5ms DropTail
$ns duplex-link $n5 $n6 16Mb 10ms DropTail
$ns duplex-link $n6 $n7 10Mb 20ms DropTail
$ns duplex-link $n6 $n8 12Mb 8ms DropTail
```

```
$ns duplex-link $n6 $n9 10Mb 8ms DropTail
$ns duplex-link $n10 $n2 10Mb 8ms DropTail
$ns duplex-link $n11 $n2 10Mb 8ms DropTail
$ns duplex-link $n12 $n2 10Mb 8ms DropTail
$ns duplex-link $n13 $n2 10Mb 8ms DropTail
$ns duplex-link $n14 $n2 10Mb 8ms DropTail
$ns duplex-link $n1 $n2 10Mb 8ms DropTail
$ns duplex-link $n2 $n6 12Mb 8ms DropTail
$ns duplex-link $n2 $n3 12Mb 8ms DropTail
$ns duplex-link $n2 $n4 10Mb 8ms DropTail
$ns duplex-link $n4 $n5 20Mb 5ms DropTail
$ns duplex-link $n3 $n6 18Mb 5ms DropTail
$ns duplex-link $n5 $n6 16Mb 10ms DropTail
$ns duplex-link $n6 $n7 10Mb 20ms DropTail
$ns duplex-link $n6 $n8 12Mb 8ms DropTail
$ns duplex-link $n6 $n9 10Mb 8ms DropTail
$ns duplex-link $n10 $n2 10Mb 8ms DropTail
$ns duplex-link $n11 $n2 10Mb 8ms DropTail
$ns duplex-link $n12 $n2 10Mb 8ms DropTail
$ns duplex-link $n13 $n2 10Mb 8ms DropTail
$ns duplex-link $n14 $n2 10Mb 8ms DropTail
# Ajustando os monitores de Fila
$ns duplex-link-op $n1 $n2 queuePos 0.5
$ns duplex-link-op $n2 $n6 queuePos 0.5
$ns duplex-link-op $n2 $n3 queuePos 0.5
$ns duplex-link-op $n2 $n4 queuePos 0.5
$ns duplex-link-op $n4 $n5 queuePos 0.5
$ns duplex-link-op $n3 $n6 queuePos 0.5
$ns duplex-link-op $n5 $n6 queuePos 0.5
$ns duplex-link-op $n6 $n7 queuePos 0.5
$ns duplex-link-op $n6 $n8 queuePos 0.5
$ns duplex-link-op $n6 $n9 queuePos 0.5
$ns duplex-link-op $n10 $n2 queuePos 0.5
$ns duplex-link-op $n11 $n2 queuePos 0.5
```

```
$ns duplex-link-op $n12 $n2 queuePos 0.5
$ns duplex-link-op $n13 $n2 queuePos 0.5
$ns duplex-link-op $n14 $n2 queuePos 0.5
# Ajustando o protocolo de roteamento padrão
$ns rtproto DV
# Definindo o trafego
# Criando duas conexões UDPs
# Primeira conexão UDP0 entre os nós 0 e 7
set udp0 [new Agent/UDP]
set null0 [new Agent/Null]
$ns attach-agent $n0 $udp0
$ns attach-agent $n7 $null0
$ns connect $udp0 $null0
$udp0 set class_ 1
# Inserindo tráfego CBR (áudio) na primeira conexão UDP0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 375
$cbr0 set interval_ 0.003
$cbr0 attach-agent $udp0
# Segunda conexão UDP1 entre os nós 1 e 8
set udp1 [new Agent/UDP]
set null1 [new Agent/Null]
$ns attach-agent $n1 $udp1
$ns attach-agent $n8 $null1
$ns connect $udp1 $null1
$udp1 set class_ 3
# Inserindo tráfego CBR (video) na segunda conexão UDP1
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 375
$cbr1 set interval_ 0.003
$cbr1 attach-agent $udp1
#Criando uma conexão TCP entre os nós 10 e 9
# e ajustando o tamanho do pacote para 1000, padrão do ns
set tcp0 [new Agent/TCP]
```

```
$tcp0 set packetSize_ 1000
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n10 $tcp0
$ns attach-agent $n9 $tcpsink0
$ns connect $tcp0 $tcpsink0
$tcp0 set class_ 2
# Inserindo tráfego FTP na conexão TCP
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
#Criando quatro conexões para tráfego de backgraund
#Primeira conexão TCP1 entre os nós 11 e 6
set tcp1 [new Agent/TCP]
$tcp1 set packetSize_ 1000
set tcpsink1 [new Agent/TCPSink]
$ns attach-agent $n11 $tcp1
$ns attach-agent $n6 $tcpsink1
$ns connect $tcp1 $tcpsink1
$tcp1 set class_ 2
# Inserindo tráfego FTP na conexão TCP1
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
#Segunda conexão TCP2 entre os nós 12 e 6
set tcp2 [new Agent/TCP]
$tcp2 set packetSize_ 1000
set tcpsink2 [new Agent/TCPSink]
$ns attach-agent $n12 $tcp2
$ns attach-agent $n6 $tcpsink2
$ns connect $tcp2 $tcpsink2
$tcp2 set class_ 2
# Inserindo tráfego FTP na conexão TCP2
set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
#Terceira conexão TCP3 entre os nós 13 e 6
set tcp3 [new Agent/TCP]
```

```
$tcp3 set packetSize_ 1000
set tcpsink3 [new Agent/TCPSink]
$ns attach-agent $n13 $tcp3
$ns attach-agent $n6 $tcpsink3
$ns connect $tcp3 $tcpsink3
$tcp3 set class_ 2
# Inserindo tráfego FTP na conexão TCP3
set ftp3 [new Application/FTP]
$ftp3 attach-agent $tcp3
#Quarta conexão TCP4 entre os nós 14 e 6
set tcp4 [new Agent/TCP]
$tcp4 set packetSize_ 1000
set tcpsink4 [new Agent/TCPSink]
$ns attach-agent $n14 $tcp4
$ns attach-agent $n6 $tcpsink4
$ns connect $tcp4 $tcpsink4
$tcp4 set class_ 2
# Inserindo tráfego FTP na conexão TCP4
set ftp4 [new Application/FTP]
$ftp4 attach-agent $tcp4
#Quarta conexão TCP5 entre os nós 11 e 6
set tcp5 [new Agent/TCP]
$tcp5 set packetSize_ 1000
set tcpsink5 [new Agent/TCPSink]
$ns attach-agent $n11 $tcp5
$ns attach-agent $n6 $tcpsink5
$ns connect $tcp5 $tcpsink5
$tcp5 set class_ 2
# Inserindo tráfego FTP na conexão TCP5
set ftp5 [new Application/FTP]
$ftp5 attach-agent $tcp5
#Quarta conexão TCP6 entre os nós 12 e 6
set tcp6 [new Agent/TCP]
$tcp6 set packetSize_ 1000
```

```
set tcpsink6 [new Agent/TCPSink]
$ns attach-agent $n12 $tcp6
$ns attach-agent $n6 $tcpsink6
$ns connect $tcp6 $tcpsink6
$tcp6 set class_ 2
# Inserindo tráfego FTP na conexão TCP6
set ftp6 [new Application/FTP]
$ftp6 attach-agent $tcp6
#Quarta conexão TCP7 entre os nós 13 e 6
set tcp7 [new Agent/TCP]
$tcp7 set packetSize_ 1000
set tcpsink7 [new Agent/TCPSink]
$ns attach-agent $n13 $tcp7
$ns attach-agent $n6 $tcpsink7
$ns connect $tcp7 $tcpsink7
$tcp7 set class_ 2
# Inserindo tráfego FTP na conexão TCP7
set ftp7 [new Application/FTP]
$ftp7 attach-agent $tcp7
#Quarta conexão TCP8 entre os nós 14 e 6
set tcp8 [new Agent/TCP]
$tcp8 set packetSize_ 1000
set tcpsink8 [new Agent/TCPSink]
$ns attach-agent $n14 $tcp8
$ns attach-agent $n6 $tcpsink8
$ns connect $tcp8 $tcpsink8
$tcp8 set class_ 2
# Inserindo tráfego FTP na conexão TCP8
set ftp8 [new Application/FTP]
$ftp8 attach-agent $tcp8
# Definindo o procedimento final
proc finish {} {
global ns ni nil
$ns flush-trace
```

```
close $ni
close $nil
exec nam redeip.nam &
exit 0
}
#Definindo Tempo de Simulação
set testTime 60.0
# Programando a simulação
$ns at 0.05 "$cbr0 start"
$ns at 0.05 "$cbr1 start"
$ns at 0.1 "$ftp0 start"
$ns at 0.1 "$ftp1 start"
$ns at 0.1 "$ftp2 start"
$ns at 0.1 "$ftp3 start"
$ns at 0.1 "$ftp4 start"
$ns at 0.2 "$ftp5 start"
$ns at 0.2 "$ftp6 start"
$ns at 0.2 "$ftp7 start"
$ns at 0.2 "$ftp8 start"
$ns at testTime "$ftp5 stop"
$ns at testTime "$ftp6 stop"
$ns at testTime "$ftp7 stop"
$ns at testTime "$ftp8 stop"
$ns at testTime "$ftp1 stop"
$ns at testTime "$ftp2 stop"
$ns at testTime "$ftp3 stop"
$ns at testTime "$ftp4 stop"
$ns at testTime "$cbr0 stop"
$ns at testTime "$cbr1 stop"
# Chamando procedimento final após warm up segundos do tempo de
simulação
$ns at [expr $testTime + 0.1] "finish"
# Iniciando a simulação
$ns run
```

ANEXO B

```
# Autor: Daiane Vaz Lima.
# Data: Março de 2007.
# Nota: Rede  MPLS

set ns [new Simulator]
set cir0  1000000
set rate0 8000000
set testTime 60.0
set packetSize 1000
#
#Definicao das cores dos fluxos de dados
$ns ldp-request-color green
$ns ldp-mapping-color red
$ns ldp-withdraw-color magenta
$ns ldp-release-color orange
$ns ldp-notification-color yellow
# Ajustando cores para animacao no nam
#Cor do pacote CBR (voz)
$ns color 1 red
#Cor do pacote CBR (video)
$ns color 3 green
#Cor dos pacotes FTPs
$ns color 2 blue

# Arquivo de trace
set nm [open redemplsp.nam w]
$ns namtrace-all $nm
# Abrindo parametros do arquivo de trace
set nml [open logmplsp.tr w]
$ns trace-all $nml
#
```

```
# ------------ Criacao da topologia da rede (inicio) ----------------------------------
# Criando os Nos
set n0 [$ns node]
set n1 [$ns node]
$ns node-config -MPLS ON
set LSR2 [$ns node]
set LSR3 [$ns node]
set LSR4 [$ns node]
set LSR5 [$ns node]
set LSR6 [$ns node]
$ns node-config -MPLS OFF
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]
set n11 [$ns node]
set n12 [$ns node]
set n13 [$ns node]
set n14 [$ns node]
# Criando os rótulos para os Nós
$n0 label "Origem_Áudio"
$n1 label "Origem_Video"
$LSR2 label "SP_LSR"
$LSR3 label "DF_LSR"
$LSR4 label "BH_LSR"
$LSR5 label "UNICEUB_LSR"
$LSR6 label "RJ_LSR"
$n7 label "Destino_Audio"
$n8 label "Destino_Video"
$n9 label "BackG"
$n10 label "BackG"
$n11 label "BackG"
$n12 label "BackG"
$n13 label "BackG"
```

```
$n14 label "BackG"
# Criando os Links
$ns duplex-link $n1 $LSR2 10Mb 8ms DropTail
$ns duplex-link $LSR2 $LSR6 12Mb 8ms DropTail
$ns duplex-link $LSR2 $LSR3 12Mb 8ms DropTail
$ns duplex-link $LSR2 $LSR4 10Mb 8ms DropTail
$ns duplex-link $LSR4 $LSR5 20Mb 5ms DropTail
$ns duplex-link $LSR3 $LSR6 18Mb 5ms DropTail
$ns duplex-link $LSR5 $LSR6 16Mb 10ms DropTail
$ns duplex-link $LSR6 $n7 10Mb 20ms DropTail
$ns duplex-link $LSR6 $n8 12Mb 8ms DropTail
$ns duplex-link $LSR6 $n9 10Mb 8ms DropTail
$ns duplex-link $n10 $LSR2 10Mb 8ms DropTail
$ns duplex-link $n11 $LSR2 10Mb 8ms DropTail
$ns duplex-link $n12 $LSR2 10Mb 8ms DropTail
$ns duplex-link $n13 $LSR2 10Mb 8ms DropTail
$ns duplex-link $n14 $LSR2 10Mb 8ms DropTail
# Ajustando os monitores de Fila
$ns duplex-link-op $n1 $LSR2 queuePos 0.5
$ns duplex-link-op $LSR2 $LSR6 queuePos 0.5
$ns duplex-link-op $LSR2 $LSR3 queuePos 0.5
$ns duplex-link-op $LSR2 $LSR4 queuePos 0.5
$ns duplex-link-op $LSR4 $LSR5 queuePos 0.5
$ns duplex-link-op $LSR3 $LSR6 queuePos 0.5
$ns duplex-link-op $LSR5 $LSR6 queuePos 0.5
$ns duplex-link-op $LSR6 $n7 queuePos 0.5
$ns duplex-link-op $LSR6 $n8 queuePos 0.5
$ns duplex-link-op $LSR6 $n9 queuePos 0.5
$ns duplex-link-op $n10 $LSR2 queuePos 0.5
$ns duplex-link-op $n11 $LSR2 queuePos 0.5
$ns duplex-link-op $n12 $LSR2 queuePos 0.5
$ns duplex-link-op $n13 $LSR2 queuePos 0.5
$ns duplex-link-op $n14 $LSR2 queuePos 0.5
#Configurando mensagens ldps em todos os nos MPLS
```

```
for { set i 2} {$i < 7} {incr i} {
for {set j [expr $i+1]} {$j < 7} {incr j} {
set a LSR$i
set b LSR$j
eval $ns LDP-peer $$a $$b
}
}
# Ajustando eventos LDPs
Classifier/Addr/MPLS set control_driven_ 1
Classifier/Addr/MPLS enable-on-demand
Classifier/Addr/MPLS enable-ordered-control
# Ajustando o protocolo de roteamento padrao
$ns rtproto DV
# Definindo o trafego
# Criando duas conexões UDPs
# Conexão UDP0 entre os nós 0 e 7
set udp0 [new Agent/UDP]
set null0 [new Agent/Null]
$ns attach-agent $n0 $udp0
$ns attach-agent $n7 $null0
$ns connect $udp0 $null0
$udp0 set class_ 1
# Inserindo tráfego CBR (audio) na primeira conexão UDP0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 375
$cbr0 set interval_ 0.003
$cbr0 attach-agent $udp0
# Conexão UDP1 entre os nós 1 e 8
set udp1 [new Agent/UDP]
set null1 [new Agent/Null]
$ns attach-agent $n1 $udp1
$ns attach-agent $n8 $null1
$ns connect $udp1 $null1
$udp1 set class_ 3
```

```
# Inserindo tráfego CBR (video) na segunda conexão UDP1
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 375
$cbr1 set interval_ 0.003
$cbr1 attach-agent $udp1
#Criando a primeira conexão TCP entre os nós 10 e 6
# e ajustando o tamanho do pacote para 1000, padrão do ns
set tcp0 [new Agent/TCP]
$tcp0 set packetSize_ 1000
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n10 $tcp0
$ns attach-agent $LSR6 $tcpsink0
$ns connect $tcp0 $tcpsink0
$tcp0 set class_ 2
# Inserindo tráfego FTP na conexão TCP
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
#Segunda conexão TCP1 entre os nós 11 e 6
set tcp1 [new Agent/TCP]
$tcp1 set packetSize_ 1000
set tcpsink1 [new Agent/TCPSink]
$ns attach-agent $n11 $tcp1
$ns attach-agent $LSR6 $tcpsink1
$ns connect $tcp1 $tcpsink1
$tcp1 set class_ 2
# Inserindo tráfego FTP na conexão TCP1
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
#Terceira conexão TCP2 entre os nós 12 e 6
set tcp2 [new Agent/TCP]
$tcp2 set packetSize_ 1000
set tcpsink2 [new Agent/TCPSink]
$ns attach-agent $n12 $tcp2
$ns attach-agent $LSR6 $tcpsink2
```

```
$ns connect $tcp2 $tcpsink2
$tcp2 set class_ 2
# Inserindo tráfego FTP na conexão TCP2
set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
#Quarta conexão TCP3 entre os nós 13 e 6
set tcp3 [new Agent/TCP]
$tcp3 set packetSize_ 1000
set tcpsink3 [new Agent/TCPSink]
$ns attach-agent $n13 $tcp3
$ns attach-agent $LSR6 $tcpsink3
$ns connect $tcp3 $tcpsink3
$tcp3 set class_ 2
# Inserindo tráfego FTP na conexão TCP3
set ftp3 [new Application/FTP]
$ftp3 attach-agent $tcp3
#Quinta conexão TCP4 entre os nós 14 e 6
set tcp4 [new Agent/TCP]
$tcp4 set packetSize_ 1000
set tcpsink4 [new Agent/TCPSink]
$ns attach-agent $n14 $tcp4
$ns attach-agent $LSR6 $tcpsink4
$ns connect $tcp4 $tcpsink4
$tcp4 set class_ 2
# Inserindo tráfego FTP na conexão TCP4
set ftp4 [new Application/FTP]
$ftp4 attach-agent $tcp4
#Sexta conexão TCP5 entre os nós 11 e 6
set tcp5 [new Agent/TCP]
$tcp5 set packetSize_ 1000
set tcpsink5 [new Agent/TCPSink]
$ns attach-agent $n11 $tcp5
$ns attach-agent $LSR6 $tcpsink5
$ns connect $tcp5 $tcpsink5
```

```
$tcp5 set class_ 2
# Inserindo tráfego FTP na conexão TCP5
set ftp5 [new Application/FTP]
$ftp5 attach-agent $tcp5
#Sétima conexão TCP6 entre os nós 12 e 6
set tcp6 [new Agent/TCP]
$tcp6 set packetSize_ 1000
set tcpsink6 [new Agent/TCPSink]
$ns attach-agent $n12 $tcp6
$ns attach-agent $LSR6 $tcpsink6
$ns connect $tcp6 $tcpsink6
$tcp6 set class_ 2
# Inserindo tráfego FTP na conexão TCP6
set ftp6 [new Application/FTP]
$ftp6 attach-agent $tcp6
#Oitava conexão TCP7 entre os nós 13 e 6
set tcp7 [new Agent/TCP]
$tcp7 set packetSize_ 1000
set tcpsink7 [new Agent/TCPSink]
$ns attach-agent $n13 $tcp7
$ns attach-agent $LSR6 $tcpsink7
$ns connect $tcp7 $tcpsink7
$tcp7 set class_ 2
# Inserindo tráfego FTP na conexão TCP7
set ftp7 [new Application/FTP]
$ftp7 attach-agent $tcp7
#Nona conexão TCP8 entre os nós 14 e 6
set tcp8 [new Agent/TCP]
$tcp8 set packetSize_ 1000
set tcpsink8 [new Agent/TCPSink]
$ns attach-agent $n14 $tcp8
$ns attach-agent $LSR6 $tcpsink8
$ns connect $tcp8 $tcpsink8
$tcp8 set class_ 2
```

```
# Inserindo tráfego FTP na conexão TCP8
set ftp8 [new Application/FTP]
$ftp8 attach-agent $tcp8
# Definindo o procedimento final
proc finish {} {
global ns nm nml
$ns flush-trace
close $nm
close $nml
exec nam redemplsp.nam &
exit 0
}
#Definindo Tempo de Simulacao
set testTime 60.0
# Programando a simulacao
$ns at 0.05 "$cbr0 start"
$ns at 0.05 "$cbr1 start"
$ns at 0.1 "$ftp0 start"
$ns at 0.1 "$ftp1 start"
$ns at 0.1 "$ftp2 start"
$ns at 0.1 "$ftp3 start"
$ns at 0.1 "$ftp4 start"
$ns at 0.2 "$ftp5 start"
$ns at 0.2 "$ftp6 start"
$ns at 0.2 "$ftp7 start"
$ns at 0.2 "$ftp8 start"
$ns at testTime "$ftp5 stop"
$ns at testTime "$ftp6 stop"
$ns at testTime "$ftp7 stop"
$ns at testTime "$ftp8 stop"
# LSP para o trafego cbr0 voz
$ns at 0.15 "[$LSR6 get-module MPLS] ldp-trigger-by-withdraw 7 -1"
$ns at 0.1 "[$LSR2 get-module MPLS] make-explicit-route 6 2_3_6 3000 -1"
$ns at 0.25 "[$LSR2 get-module MPLS] flow-erlsp-install 7 -1 3000"
```

```
# LSP para o trafego cbr1 video
$ns at 0.15 "[$LSR6 get-module MPLS] ldp-trigger-by-withdraw 8 -1"
$ns at 0.1 "[$LSR2 get-module MPLS] make-explicit-route 6 2_4_5_6 3001 -1"
$ns at 0.25 "[$LSR2 get-module MPLS] flow-erlsp-install 8 -1 3001"
$ns at 2.0 "[$LSR2 get-module MPLS] ldp-trigger-by-release 7 3000"
$ns at 2.0 "[$LSR2 get-module MPLS] ldp-trigger-by-release 8 3001"
$ns at testTime "$ftp0 stop"
$ns at testTime "$ftp1 stop"
$ns at testTime "$ftp2 stop"
$ns at testTime "$ftp3 stop"
$ns at testTime "$ftp4 stop"
$ns at testTime "$cbr0 stop"
$ns at testTime "$cbr1 stop"
# # Chamando procedimento final após warm up segundos do tempo de
simulação
$ns at [expr $testTime + 1.0] "finish"
# Iniciando a simulacao
$ns run
```