FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Tecnologia em Segurança da Informação

Ely Edson Dalbem

QUALIDADE DE SERVIÇO EM REDES IPv6 COM TRATAMENTO JUSTO DE FLUXOS TCP E UDP

Americana, SP

2015

FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Tecnologia em Segurança da Informação

Ely Edson Dalbem

QUALIDADE DE SERVIÇO EM REDES IPv6 COM TRATAMENTO JUSTO DE FLUXOS TCP E UDP

Trabalho de Conclusão de Curso em cumprimento à exigência curricular do Curso de Segurança da Informação, sob orientação do Prof. Me. Rossano Pablo Pinto.

Área de concentração: Redes de Computadores.

Americana, SP

2015

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana - CEETEPS

Dados Internacionais de Catalogação-na-fonte

Dalbem, Ely Edson

D139q Qualidade de serviço em redes IPv6 com tratamento justo de fluxos TCP e UDP. / Ely Edson Dalbem. – Americana: 2015.

47f.

Monografia (Graduação em Tecnologia de Segurança da Informação). - - Faculdade de Tecnologia de Americana – Centro Estadual de Educação Tecnológica Paula Souza.

Orientador: Prof. Me. Rossano Pablo Pinto

1. Redes de computadores I. Pinto, Rossano Pablo II. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana.

CDU: 681.519

Ely Edson Dalbem

QUALIDADE DE SERVIÇO EM REDES IPv6 COM TRATAMENTO JUSTO DE FLUXOS TCP E UDP

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Segurança da Informação pelo CEETEPS/Faculdade de Tecnologia – FATEC/ Americana.
Área de concentração: Redes de Computadores.

Americana, 22 de junho de 2015.

Banca Examinadora:

Rossano Pablo Pinto (Presidente)
Mestre
Fatec de Americana-SP

Rodrigo Brito Battilana (Membro)
Graduado
Fatec de Americana-SP

Aloísio Daniel Vendemiatti (Membro)
Mestre
Fatec de Americana-SP

RESUMO

Este trabalho apresenta mecanismos para provimento de qualidade de serviço (QoS) para redes IPv6 visando oferecer tratamento justo para fluxos de rede. Para fundamentar o experimento realizado e resultados obtidos durante a elaboração deste documento, são apresentadas as características do protocolo IPv6 e os campos contidos em seu cabeçalho que dão suporte ao uso de QoS. Também são abordadas algumas arquiteturas de QoS existentes e o detalhamento da arquitetura e seu método de admissão de pacotes ao domínio *Differentiated Services* (DiffServ). A propagação de dados pela rede pode ser afetada por conta de parâmetros como o atraso de fila e perda de pacotes, assim para tratá-los são abordados os mecanismos de escalonamento *Random Early Detection* (RED) e *Stochastic Fair Blue* (SFB) oferecendo um método diferenciado para o enfileiramento de pacotes e controle de congestionamentos. Um cenário para analisar o desempenho durante a transmissão de fluxos TCP e UDP é apresentado com o uso de RED e SFB.

Palavras-chave: Redes IPv6. Mecanismos de Escalonamento de Pacotes. Qualidade de Serviço (QoS). RED, SFB, DiffServ;

ABSTRACT

This monograph presents mechanisms for the provision of Quality of Service (QoS) for IPv6 networks aiming to offer fair treatment for network flows. To support the carried out experiment and the results obtained during the development of this document, both the IPv6 features and the fields contained in its header that support the use of QoS are presented. Also, some existing QoS architectures are addressed, specially the Differentiated Services (DiffServ) architecture. The propagation of data over the network can be affected by parameters like queuing delay and packet loss. In order to deal with these problems, the scheduling mechanisms Random Early Detection (RED) and Stochastic Fair Blue (SFB) are addressed. These mechanisms offer a differentiated method for queuing packets and congestion control. A scenario to analyze the performance during the transmitting of TCP and UDP flows is presented using RED and SFB.

Keywords: IPv6 networks. Scheduling Mechanisms. Quality of Service (QoS). RED, SFB;

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1. INTRODUÇÃO

A **motivação** para a elaboração deste trabalho reflete a mudança progressiva que vem ocorrendo no mundo tecnológico com relação ao aumento do número e variedade de dispositivos conectados na rede.

Assim mediante a esta demanda origina-se o **problema** do crescimento desestruturado das redes domésticas ou empresariais de forma que toda a propagação de dados é degradada, por conta de atrasos e perdas de pacotes, de maneira que toda a intercomunicação entre dispositivos poderá ser apresentada com desempenho insatisfatório, afetando assim diretamente o consumo pelo usuário. Para reduzir tal problema, sugere-se a adição de mecanismos de QoS, a fim de que estes consigam suprir a grande demanda de transporte de dados, mitigando o impacto e incidência de perda de pacotes, limite de banda, atrasos e sua variação. Ao adotar QoS na rede é possível controlar a banda provida e tratar o tráfego gerado por aplicações para que mesmo ao atuarem em situações de congestionamento, a degradação do tráfego seja quase imperceptível ao usuário, assim permitindo a administração sobre o consumo dos recursos da rede.

As **hipóteses** que fundamentam a elaboração deste documento são:

a) O tratamento justo para fluxos mistos é possível por meio do uso de qualidade de serviço;

b) A implementação de qualidade de serviço em uma rede possibilita que os efeitos negativos de atrasos e perda de pacotes sejam mitigados / reduzidos;

c) A implantação de uma arquitetura de QoS proporciona serviços adicionais para facilitar a gestão do tráfego que necessita de tratamento preferencial.

A utilização da qualidade de serviço em redes de computadores, mostra-se importante e necessária nos dias atuais por existirem diversas aplicações sensíveis a perda de pacotes, largura de banda, atrasos e sua variação, que disputam entre si recursos da rede. Assim é preciso que haja um tratamento organizado de todo o tráfego para assegurar que funcionem corretamente em conjunto.

O **objetivo geral** deste trabalho é apresentar um modelo de estruturação de rede, no qual é oferecido aos fluxos um tratamento justo, visando reduzir a degradação do tráfego ocorrida pela perda de pacotes e por ações de mecanismos de controle de congestionamento nativos de protocolos de transporte. Desta forma o

desempenho sobre a propagação de dados é assegurada mesmo em uma rede com gargalo entre enlaces.

Já os **objetivos específicos** definidos na elaboração deste projeto são:

- Apresentar um método justo para tratamento do tráfego;

- Testar o desempenho desta arquitetura com e sem qualidade de serviço em um ambiente controlado;

- Apresentar a análise dos resultados obtidos.

É abordado um método para prover qualidade de serviço (QoS) aos tráfegos para assegurar um tratamento justo para os pacotes que utilizam o protocolo *Transmission Control Protocol* (TCP) e *User Datagram Protocol* (UDP) na camada de transporte, a fim de oferecer o controle de congestionamento por meio de mecanismos de escalonamento e assegurar o desempenho igualitário para as diversas aplicações que fazem a troca de dados via redes de computadores, indiferentemente da quantidade de carga existente nos enlaces e também demostrar como seria a reação do mesmo cenário se não houvesse a implantação de uma arquitetura de QoS.

Para obter tais resultados foram usados os **procedimentos metodológicos** comparativo entre os mecanismos de escalonamento *Random Early Detection* (RED) e *Stochastic Fair Blue* (SFB) demonstrando qual o desempenho de ambos durante o tratamento de fluxos TCP e UDP, além de um estudo de caso aplicado sobre um cenário controlado, estruturado sobre a arquitetura *Differentiated Services* (DiffServ).

Desta forma surge a seguinte **pergunta**: é possível prover qualidade de serviço a tráfegos TCP e UDP de maneira justa? Esta questão é respondida ao término deste trabalho, fundamentando tal resposta por meio dos resultados obtidos com os experimentos.

No decorrer deste documento é apresentado no Capítulo 2 a definição de Qualidade de Serviço (QoS) e uma breve abordagem sobre a sexta versão do Internet *Protocol* (IPv6) e sobre algumas arquiteturas utilizadas no mercado para redes com QoS. O Capítulo 3 apresenta os mecanismos de escalonamento RED e SFB que serão adotados para realização dos testes, explicando sobre seu funcionamento e implementação.

No quarto Capítulo é apresentado o cenário dos experimentos, descrevendo as aplicações escolhidas, o modo que funciona a rede virtualizada, a infraestrutura e

os requisitos de cada dispositivo que estarão conectados. Os métodos de como foram analisados os dados coletados são apresentados no quinto capítulo, juntamente com a demonstração gráfica dos resultados obtidos.

Por fim é demonstrado nas Considerações Finais, o conhecimento obtido com a elaboração deste documento e a ótica do autor sobre os resultados alcançados, de forma que o leitor possa avaliar as experiências vividas durante a realização deste trabalho.

2. QUALIDADE DE SERVIÇO (QOS) EM REDES IPV6

Atualmente as redes de computadores transportam uma elevada carga de fluxos contínuos, os quais possuem sensibilidade a largura de banda, a atrasos e a sua variação (*Jitter*), necessitando assim de um tratamento adequado para que a qualidade da transmissão e recepção não sejam degradadas, de forma que os impactos sofridos durante sua trajetória sejam pouco ou não perceptíveis ao usuário final no momento do consumo dos fluxos. A fim de prover os recursos solicitados para atender a demanda de uma aplicação sensível a qualidade da rede, sugere-se a implementação de uma arquitetura e mecanismos que possam oferecer serviços para garantir a qualidade de envio e recepção destes dados, assim fazendo com que a rede atue com qualidade de serviço (QoS).

2.1 Qualidade de Serviço (QoS)

QoS é definida pela Cisco (2005), como um modo para medir a disponibilidade do serviço e a qualidade de transmissão sobre uma rede computacional ou em um grupo delas, sendo esta última mensurada conforme os parâmetros existentes na rede e que afetam a trajetória dos dados de um ponto a outro.

Segundo Stallings (2005), qualidade de serviço refere-se ao desempenho obtido por um fluxo de dados ao ser submetido aos parâmetros existentes na rede, apontando o percurso a ser seguido, a prioridade de cada pacote e os recursos que o próximo roteador deverá prover a estes dados.

O tratamento do tráfego pode ser feito pela adoção de arquiteturas e mecanismos que ofereçam serviços adicionais ao modelo de melhor esforço usado por padrão em redes TCP/IP, a fim de prover os recursos exigidos para o desempenho de aplicações, priorização de determinados dados em roteadores ou controle de largura de banda, perda de pacotes, atrasos e suas variações. Cada rede possui suas peculiaridades, sendo aconselhável a avaliação de necessidades, tipos de aplicações que serão usadas e sua respectiva criticidade, a fim de definir o modelo mais adequado para prover QoS.

Alguns dos parâmetros incidentes em redes que trabalham sobre o Internet *Protocol* (IP), conforme descritos por Kurose e Ross (2010), são: o atraso de

processamento, sendo este o tempo gasto para que o cabeçalho do pacote seja analisado em busca de corrupção de dados; atraso de fila, que refere-se ao tempo em que o pacote permanece enfileirado esperando para ser transmitido; o atraso de propagação, o qual representa o período em que o datagrama percorre um enlace até o próximo nó; e o atraso de transmissão, que representa o tempo gasto para que o roteador envie os bits de um pacote da fila para a porta correspondente ao caminho a ser seguido. A somatória destes elementos, presentes entre origem e destino é conhecida por atraso fim-a-fim.

Além destes parâmetros existem alguns outros: como a taxa de erros, que representa a corrupção binária dos dados, a qual pode resultar desde uma troca de caracteres em um arquivo até a falha de execução de uma aplicação que dependia destes dados; a perda de pacotes, que afeta a entrega ao destino, de forma que em grande escala pode ocasionar dificuldade ou término de comunicação entre pontos ou até mesmo gerar congestionamento na rede por conta de reenvios dos dados perdidos; e também a variação de atrasos, conhecida como *jitter*, o qual a CISCO (2005) descreve como a diferença do valor de atraso fim-a-fim entre pacotes do mesmo fluxo e por Tanenbaum (2003) como a entrega de pacotes de maneira não uniforme, de forma que algumas aplicações multimídia precisem adotar mecanismos para organizá-los.

Outro parâmetro é a vazão (*throughput*), que representa a quantidade de pacotes que podem ser encaminhados em um intervalo de tempo, ou seja, a largura disponível para transporte de dados pelo enlace, de forma que quanto maior for a banda disponível mais bits serão transmitidos por segundo e ao entrar em um enlace de menor capacidade poderá haver a redução de dados para adequação (FENG *et al,* 2002).

Todos estes parâmetros podem incidir durante a propagação de um fluxo de modo que o mesmo poderá ser afetado negativamente se não houver métodos para controlá-los, assim podendo comprometer o desempenho de uma aplicação e proporcionar ao usuário uma sensação negativa durante a utilização. Desta forma a experimentação da aplicação resultará em avaliação subjetiva feita pelo usuário, a qual é definida como qualidade de experiência (QoE) (PATRICK *et al,* 2004).

2.2 Internet *Protocol*

A Internet utiliza hoje o Internet *protocol* versão 4 (IPv4) por este ser um protocolo amplamente difundido e conhecido pelo mercado, porém a versão 4 do protocolo possui limitação de endereços válidos, o que ocasionou recentemente o esgotamento da oferta desses endereços pelas entidades reguladoras (SANTOS *et al,* 2011).

Já em 1992, de acordo com Das (2008), notou-se a grande possibilidade do esgotamento de endereços IPv4. Sendo assim foi desenvolvido a sexta versão do Internet *protocol* (IPv6) a fim de suprir as necessidades mercadológicas, oferecendo uma maior quantidade de endereços disponíveis que são estruturadas usando oito quartetos hexadecimais divididos por pontuação dupla, resultando em um endereço de 128 bits (seu antecessor utiliza 32 bits).

A mudança entre as versões 4 e 6 está ocorrendo lentamente, estando em maio de 2015 o Estados Unidos com 15,05%, a Alemanha com 14,82% e o Peru com 13,58% de adoção no país, sendo que os demais países possuem índices inferiores a 10%, conforme as estatísticas apresentadas no sítio do Google[1]. Durante este período transitório o uso de técnicas como o tunelamento, pilha dupla e tradução se farão necessários para permitir tal compatibilidade, conforme descrito por Santos *et al* (2011). O tunelamento e a pilha dupla são descritas na RFC 4213, sendo o tunelamento um método que encapsula um datagrama IPv6 em IPv4 e a pilha dupla trabalha com equipamentos que suportam ambas versões do protocolo IP, adequando o pacote conforme o endereçamento do enlace, porém exige que exista um servidor *Domain Name System* (DNS) para resolver os endereços (FOROUZAN, 2008; NORDMARK; GILLIGAN, 2005). Já a técnica de tradução converte os campos do cabeçalho para um formato inteligível pelo nó, de forma que este seja capaz de processar o datagrama.

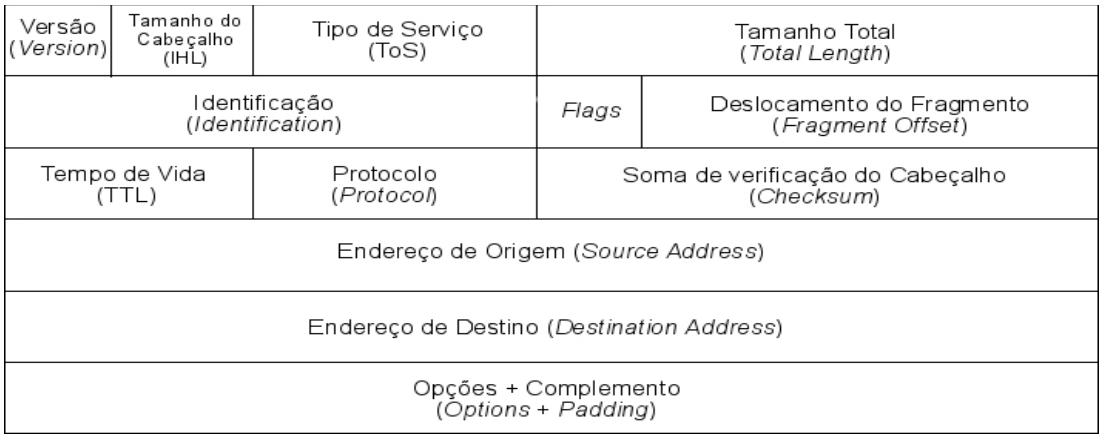
A versão 4 do Internet *Protocol*, possui um cabeçalho com tamanho que varia de 20 a 60 bytes. Os 20 primeiros bytes representam os doze campos fixos. O campo opções pode aumentar o cabeçalho para até 60 bytes. O campo Internet *Header Length* (IHL) é composto por 4 bits, o qual é responsável por armazenar a quantidade de *words* de 32 bits que representa o tamanho total do cabeçalho IPv4,

1 Informações disponíveis no sítio www.google.com/intl/en/ipv6/statistics.html

sendo o valor mínimo deste 5 e seu limite máximo 15, ambos em decimal (SANTOS *et al,* 2011; TANENBAUM, 2003).

Já o campo *Type of Service* (ToS) contido no cabeçalho do protocolo IPv4 composto por 8 bits, tem sua função descrita por Kurose e Ross (2010) como meio para distinção de pacotes com necessidade de QoS dos não sensíveis aos parâmetros da rede, de forma que um nó possa identificá-lo e oferecer tratamento diferenciado a este conforme o tipo definido para o valor marcado.

Figura 1: Cabeçalho IPv4



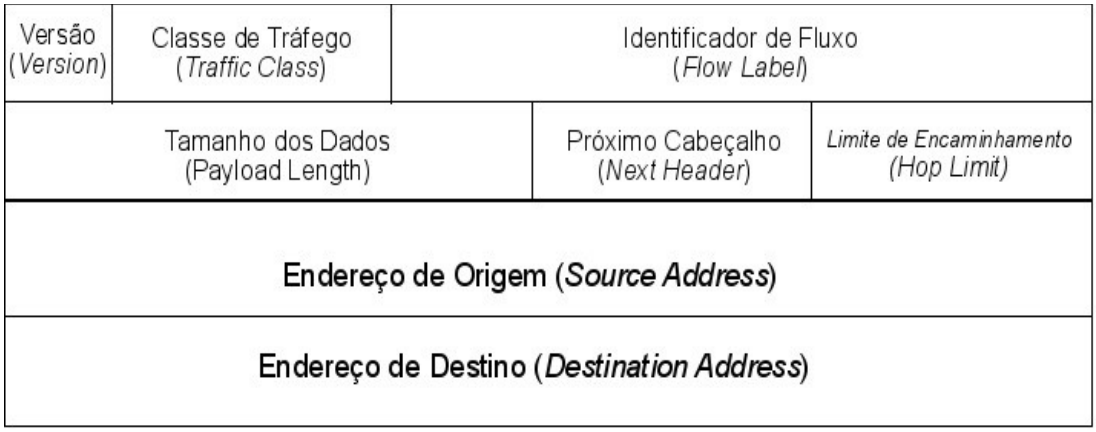
Fonte: DEGERMARK *et al* (1999, p. 33); TANENBAUM (2003, p. 461)

Além destes campos o protocolo IP em sua quarta versão apresenta mais dez fixos e outro opcional, conforme é mostrado na Figura 1. O campo *Options* não é fixo, desta forma podendo complementar as informações contidas no cabeçalho, podendo ser usado para apresentar o grau de segurança do pacote, armazenar o endereço de roteadores durante sua propagação, registrar o horário que foi processado por um nó, entre outras opções (KUROSE; ROSS, 2010; TANENBAUM, 2003).

Na sexta versão do protocolo IP alguns campos que existiam no cabeçalho de seu antecessor IPv4 foram removidos com intuito de simplificar sua formatação, assim o cabeçalho IPv6 foi estruturado em 8 campos fixos além de permitir o uso de cabeçalhos de extensão para aprimorar suas informações, conforme apresentado na Figura 2. Houve também a renomeação dos campos *Type of Service* (ToS), *Total Length*, *Time to Live* (TTL) e *Protocol* para *Traffic Class*, *Payload Length*, *Hop Limit* e *Next Header* respectivamente, além de serem realocados no cabeçalho (DEERING, 1998; SANTOS *et al,* 2011). Foi inserido o campo *Flow Label*, descrito pela equipe

do Núcleo de Informação e Coordenação do Ponto BR (SANTOS *et al,* 2011) como identificador de datagramas pertencentes a um determinado fluxo, oferecendo um campo adicional para sinalização de fluxos com qualidade de serviço.

Figura 2: Cabeçalho IPv6



Fonte: SANTOS *et al* (2011) e TANENBAUM (2003, p. 497)

2.3 Arquiteturas de QoS

O provimento de qualidade de serviço ao tráfego em uma rede, pode ser alcançado com a adoção de algum modelo de arquitetura em conjunto com mecanismos e técnicas que forneçam serviços extras, de forma que seja possível atender a necessidade de recursos de determinadas aplicações, como garantia de banda, prioridade na transmissão, controle do *jitter,* entre outros, assegurando desta maneira o consumo satisfatório da mídia mesmo quando houver grande utilização do enlace ou congestionamento nos nós (PINTO *et al,* 2003; TANENBAUM, 2003).

Para que uma arquitetura atenda adequadamente as necessidades da rede, sugere-se que seja feito um levantamento das aplicações que serão usadas, a prioridade no tratamento de cada fluxo, quais os recursos disponíveis e como estes serão distribuídos. Após a apuração desta demanda é possível escolher o modelo adequado de arquitetura, serviços e mecanismos adicionais, assim montando uma estrutura que ofereça QoS. Aconselha-se também que todos os equipamentos da infraestrutura tenham sua alocação física, suas conexões e sua função dentro da rede mapeadas, para que uma falha física seja rapidamente identificada e ações sejam tomadas para que a qualidade de serviço da rede não seja afetada (DALBEM,

2014). Algumas das arquiteturas difundidas no mercado são a *Differentiated Services* (DiffServ), *Integrated Services* (IntServ) e *Multi Protocol Label Switching* (MPLS). O DiffServ efetua a marcação dos seis primeiros bits do campo *Type of Service* (ToS) no cabeçalho IPv4 ou *Traffic Class* para o IPv6, com o *Differentiated Service Code Point* (DSCP), definindo qual será a chance de descarte e prioridade deste pacote, dividida entre quatorze possibilidades de tratamento (BLAKE *et al,* 1998; FOROUZAN, 2008). Esta arquitetura será detalhada no Capítulo 3.

A arquitetura IntServ trabalha com a reserva de recursos junto aos nós tentando suprir a necessidade de cada fluxo, sendo tal ação feita pelo *Resource Reservation Protocol* (RSVP), o qual inicialmente estabelece conexão com o destinatário, enviando em seguida um pacote PATH para os roteadores que compõe o trajeto solicitando que atendam as exigências de determinado fluxo, de forma que estes poderão responder com a mensagem RESV confirmando a alocação ou caso não disponham destes recursos enviarão RESV-ERR negando atendimento. Após concluir a transferência de dados o RSVP pedirá aos nós que cancelem a reserva por meio da mensagem RESV-TEAR e posterior PATH-TEAR para limparem as informações sobre o tráfego gerado (BRADEN *et al,* 1997; WROCLAWSKI, 1997). Esta arquitetura oferece também um algoritmo para administrar as solicitações de reserva, controlando os recursos disponíveis em cada nó e evitar que hajam pedidos abusivos que comprometerão a chance de atender mais fluxos e também o algoritmo classificador que define em qual classe o fluxo será alocado, a fim de uniformizar o provimento de recursos (BRADEN; CLARK; SHENKER, 1994). Além destes mecanismos o IntServ dispõe de serviços para garantir uma quantidade específica de banda e taxa máxima de atrasos entre a origem e o destino (RFC 2212) e o serviço de carga controlada (RFC 2211) provê tratamento diferenciado a um fluxo de forma que sua propagação ocorra como se estivesse em uma rede quase ociosa, permitindo desta maneira uma alta taxa de sucesso na entrega e com baixo índice de atrasos (SHENKER *et al,* 1997; WROCLAWSKI, 1997a).

Já a *Multi Protocol Label Switching* (MPLS) trabalha com a inserção de um rótulo entre os cabeçalhos gerados na multiplexação das camadas de rede e enlace do pacote durante seu ingresso ao domínio, agregando a este uma *Forward Equivalent Class* (FEC) a qual irá vinculá-lo a um determinado grupo de dados pelo qual ele será identificado e tratado (ROSEN *et al,* 2001). É usado o protocolo *Open Shortest Path First* (OSPF) para estabelecer a melhor rota em que o pacote seguirá,

podendo esta ser alterada durante sua trajetória, já que um o rótulo será substituído ao ser recebido pelo *Label Switched Routers* (LSR), de modo que um novo será inserido com informações que auxiliarão o nó seguinte a escolher o próximo salto, visando assim otimizar a transmissão dos dados (ROSEN *et al,* 2001; TANENBAUM, 2003).

Segundo Allman e Floyd (2008) e Stallings (2005), caso não seja adotada uma arquitetura, por padrão será aplicado o modelo *Best Effort* (BE), ou melhor esforço, que trata de maneira similar todos os pacotes, de forma que todos disputam recursos de forma justa, mas que alguns fluxos podem ser mais beneficiados do que outros, como é o caso de fluxos UDP x TCP.

2.4 Mecanismo de *Slow Start* do Protocolo TCP

O mecanismo de *Slow Start* faz parte do conjunto de algoritmos para controle de congestionamento do protocolo TCP, sendo seu foco a redução da taxa de envio de pacotes de um fluxo com posterior aumento gradativo. Para possibilitar seu funcionamento são definidas as variáveis *Congestion Window* (cwnd) e *Receiver Window* (rwnd). A variável cwnd armazena a quantidade de pacotes enviados iniciando em 1, sendo seu valor dobrado a cada confirmação de recebimento (ACK) recebida. Já a variável rwnd mantém a quantidade de dados que o destinatário pode receber por envio, sendo este valor estabelecido na etapa de conexão (ALLMAN *et al*, 2009).

O acionamento do algoritmo de *Slow Start* é feito nas situações de envio inicial, após a perda de pacotes e ao receber confirmação de recebimento (ACK) duplicada. A utilização do mecanismo de *Slow Start* no início do envio visa apurar a condição atual da rede e determinar sua capacidade para propagação dos dados. Já ao ser detectada a perda de pacote por meio do não recebimento de ACK ou por receber um ACK duplicado, será acionado o *Slow Start* para reduzir a taxa de envio de pacotes do fluxo, visando evitar o agravo do congestionamento (ABOUZEID; ROY, 2000; ALLMAN *et al,* 2009; STEVENS, 1997). Desta maneira a taxa de transmissão será reduzida podendo ocorrer uma redução global da taxa de transferência, ocasionando atrasos e desperdício de banda (ABOUZEID; ROY, 2000; ALLMAN *et al*, 2009; STEVENS, 1997).

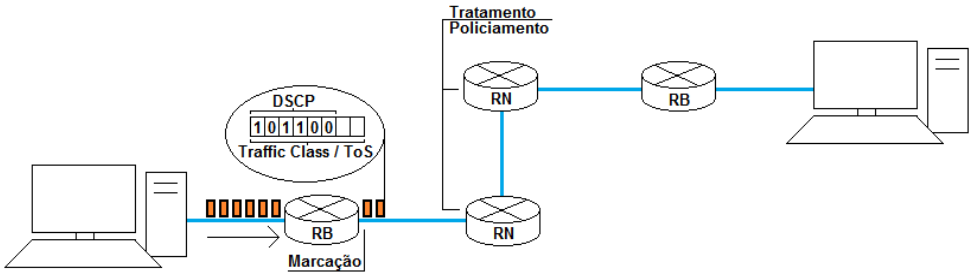
3. ARQUITETURA DIFFSERV E MECANISMOS DE ESCALONAMENTO

A implementação da arquitetura DiffServ em uma rede irá prover serviços adicionais, os quais permitirão uma melhor gestão dos fluxos com necessidade de tratamento diferenciado. A administração do enfileiramento de pacotes é feita por meio do mecanismo de escalonamento usado pelo roteador, o qual por padrão utiliza o método *First In First Out* (FIFO). FIFO organiza os pacotes em uma fila única conforme sua ordem de chegada, podendo este não ser o ideal para alguns modelos de redes que trabalham com QoS (DALBEM, 2014; KUROSE; ROSS, 2010).

3.1 DiffServ

DiffServ provê qualidade de serviço aos dados da rede por meio de tratamento diferenciado aos fluxos conforme a marcação submetida a seus pacotes.

Figura 3: Funcionamento da Arquitetura DiffServ



Fonte: Baseado em PINTO *et al* (2003, p. 333)

A Figura 3 apresenta a marcação realizada nos pacotes por meio do roteador de borda (RB), que é responsável pelo ingresso destes dados na rede. Porém existe a possibilidade de implementar de maneira simplificada a arquiteturas DiffServ, permitindo assim que a máquina origem efetue a própria marcação, usando por exemplo o software *Traffic Control*[2] (TC) nativo de sistemas GNU/Linux. Os roteadores de núcleo (RN) são encarregados de fornecer o atendimento conforme o DSCP informado no pacote, de forma que seja direcionado ao destino respeitando sua ordem de prioridade ou descartado em casos de congestionamento, de acordo com sua precedência de descarte. Estes dois tipos de roteadores também realizam

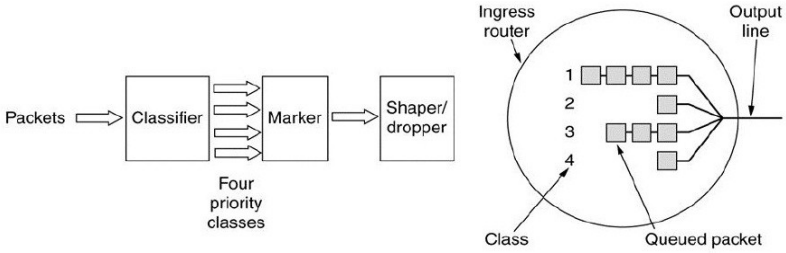
2 Informações disponíveis no sítio http://www.lartc.org

o policiamento do fluxo, a fim de analisar a marcação do pacote e checar se a mesma condiz com os recursos configurados. Caso a mesma esteja em desacordo haverá a remarcação dos pacotes visando adequar sua propagação (NICHOLS *et al,* 1998).

Para que todo o processo de troca de dados seja iniciado dentro de uma arquitetura DiffServ, os hosts e domínios de origem e destino deverão estabelecer um acordo com o intuito de definir os valores para cada parâmetro e o modelo de encaminhamento de dados que deverão ser cumpridos durante a trajetória de determinado fluxo. O acordo é denominado *Service Level Agreement* (SLA). Nas cláusulas deste acordo poderá conter o *Traffic Conditioning Agreement* (TCA), que são regras que detalham como o tráfego será condicionado e também elementos sobre o tratamento dos dados, que ao serem aceitos entre as partes ficarão mantidos no *Traffic Conditioning Specification* (TCS) juntamente com as demais especificações sobre os valores definidos para os parâmetros, os quais farão parte do *Service Level Specification* (SLS), que ditará os recursos a serem providos a um determinado fluxo conforme a marcação que este carregue (STALLINGS, 2005).

O DiffServ trabalha com os modelos de encaminhamento *Per Hop Behavior* (PHB), os quais representam quatorze níveis possíveis para marcações de pacote, divididos entre o *Assured Forwarding* (AF), que detêm doze dessas possibilidades, o *Expedited Forwardind* (EF) com uma e outra para os fluxos que não recebem marcação, tratados assim com o padrão BE, sendo os recursos oferecidos para cada uma, delimitados durante o SLA (DAVIE *et al,* 2002; GROSSMAN, 2002; HEINANEN, 1999; JACOBSON *et al,* 1999).

Figura 4: *Assured Forwarding* (AF)

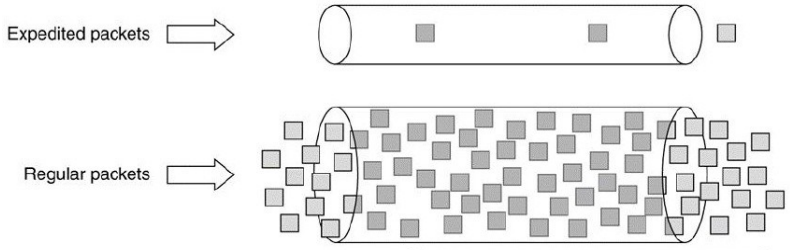


Fonte: TANENBAUM (2003, p. 441)

O modelo AF, mostrado na Figura 4, trabalha com cada nível composto por argumentos sobre a prioridade de tratamento e a precedência de descarte dos

pacotes, sendo este primeiro com quatro possibilidades descritas em ordem crescente de 1 a 4, sendo a última com menor preferência de atendimento e maior chance de sofrer atraso. Já a chance de descarte poderá ser baixa, média ou alta, escritas de 1 a 3 respectivamente, sendo a marcação binária 100110, ou AF43, considerada a pior por combinar os argumentos 4 e 3 (HEINANEN, 1999; TANEMBAUM, 2003).

Figura 5: *Expedited Forwarding* (EF)



Fonte: TANENBAUM (2003, p. 440)

Já o EF representa o nível com maior prioridade da arquitetura DiffServ, oferecendo aos fluxos com a marcação binária 101100 um tratamento como se estivessem em túnel exclusivo, de forma que não precisará disputar recursos com pacotes não marcados ou marcados por AF, já que possuirá valores alocados para atender a demanda específica dos dados desta aplicação (DAVIE *et al,* 2002; JACOBSON *et al,* 1999; TANEMBAUM, 2003). Conforme ilustrado na Figura 5.

Esta arquitetura trabalha com um gestor de recursos denominado *Bandwidth Broker* (BB), o qual é encarregado de administrar a banda a ser disponibilizada para o tráfego de dados e por garantir que a política estabelecida durante a negociação do SLA seja cumprida, de ponto a ponto e entre domínios DiffServ, de forma que a qualidade de serviço exigida possa ser provida pelos roteadores que compõe esta rede (GUNTER; BRAUN, 1999; PINTO *et al,* 2003).

Para definir quais os fluxos que serão tratados prioritariamente é necessário que exista um mecanismo para controlar a admissão na rede DiffServ, de forma que sua marcação seja correspondente as configurações evitando assim que um fluxo marcado por outra fonte, ao invés do roteador de borda que é o responsável por tal função considerando o cenário apresentado na Figura 3, utilize recursos que não seriam providos a ele. O controle de admissão pode ser realizado utilizando algum software que permite a filtragem e consiga agregar dados ao cabeçalho IP, como por

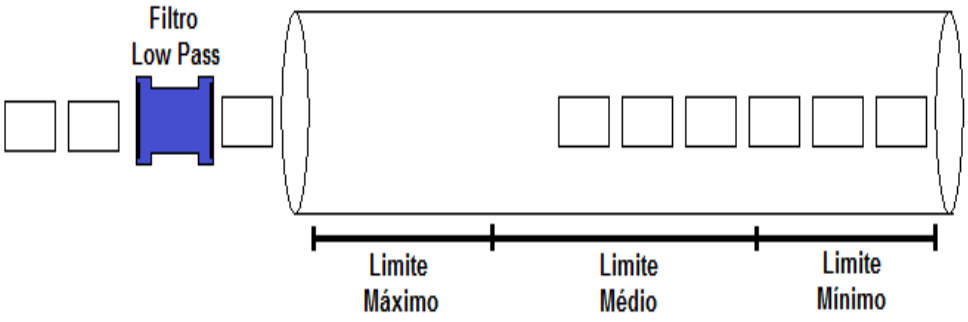
exemplo usando o firewall para filtrar tráfego *User Datagram Protocol* (UDP) e marcar seu campo DSCP.

Desta forma após a marcação dos pacotes os mesmos serão propagados pela rede, onde ao passarem pelos próximos nós serão enfileirados e receberão o tratamento previamente determinado que corresponda a marca identificada em seu campo DSCP, sendo tal organização imposta pelos algoritmos de fila, como o RED e o SFB abordados nas seções 3.2 e 3.3.

3.2 *Random Early Detection* (RED)

O *Random Early Detection* (RED), também conhecido como *Random Early Discard* ou *Random Early Drop*, foi desenvolvido para prevenir a perda de pacotes em enlaces congestionados. Para evitar que ocorra o congestionamento, tenta identificar de maneira antecipada a ocupação do enlace, tomando como base a média de crescimento da fila. Caso a média de enfileiramento exceda os valores determinados será iniciado o processo de notificação de congestionamento, feito pela marcação dos pacotes, visando assim controlar o crescimento da fila. A escolha do fluxo que receberá a marca é feita de maneira aleatória, de forma que não haja distinção por tipo de dados. Apenas o fluxo escolhido deverá reduzir o envio de pacotes, visando assim evitar que ocorra a redução geral da taxa de transferência de dados durante um momento de congestão, a qual é conhecida como sincronização global. Se o limite máximo de enfileiramento for atingido, ocorrerá o descarte de pacotes (ABOUZEID; ROY, 2000; FENG *et al,* 1999; FLOYD; JACOBSON, 1993).

Figura 6: Algoritmo de fila RED



Fonte: Baseado em ZHENG (2002).

A marcação dos pacotes é feita no campo *Explicit Congestion Notification* (ECN) representado pelos últimos dois bits dos oito existentes no campo *Traffic Class* do cabeçalho IPv6 ou *Type of Service* (ToS) no IPv4, sendo seu objetivo avisar sobre o estado de congestão da rede, a fim de possibilitar que a origem e destino tomem providências para evitar a perda de pacotes por conta de possíveis descartes (RAMAKRISHNAN *et al,* 2001).

O funcionamento do algoritmo RED demonstrado na Figura 6, se baseia na detecção do tráfego que está chegando ao roteador, de forma que seja possível tomar uma ação antecipada quanto ao congestionamento. Esta detecção é feita por meio do filtro *Low Pass*, o qual trabalha com a média exponencial ponderada para definir se o limite médio da fila (avg) será excedido (ZHENG, 2002). Assim caso seja identificado que o limite mínimo de enfileiramento, ou "*minimum threshold for queue"* (minth) foi superado, alguma conexão será escolhida de maneira aleatória, porém maior será a probabilidade de escolha de fluxos que estão utilizando uma maior largura de banda em receber a marcação, a qual tem intuito de notificar o destinatário para solicitar ao remetente que reduza a janela de envio de dados desta conexão visando evitar que o congestionamento de fato ocorra. Já caso exceda o limite máximo determinado para a fila, ou "*maximum threshold for queue"* (maxth), o RED irá marcar todos os próximos pacotes que chegarem ao roteador, de modo que apenas ocorra descarte se a conexão não reduzir sua taxa de envio, esta ação será feita até que o comprimento da fila seja diminuído, retornando ao método de marcação aleatória ou cancelando a marcação caso a situação tenha normalizado (ABOUZEID; ROY, 2000; FENG *et al,* 2002).

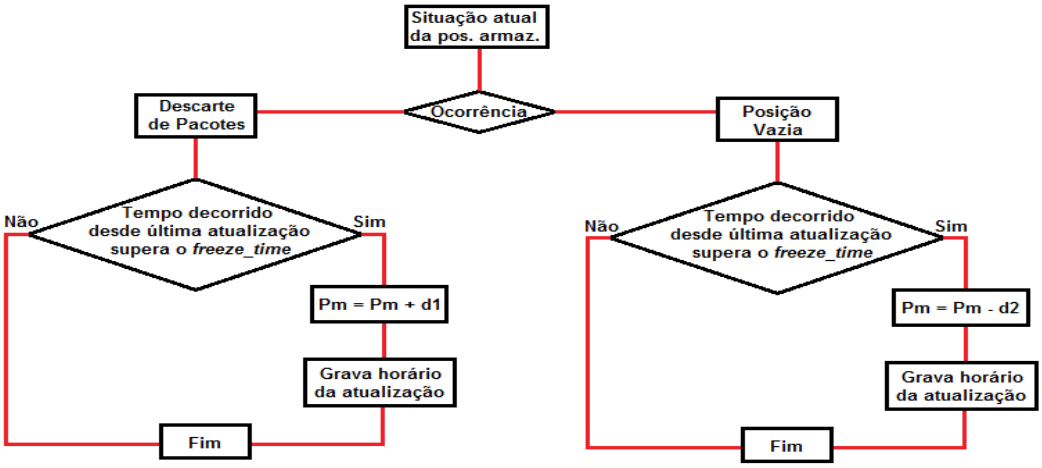
Para determinar a probabilidade de marcação (Pa) é usado o valor obtido por meio do cálculo (Pb / (1 - *count* * Pb)), sendo a variável *count* a representação da quantidade de pacotes enviados desde a última ocorrência de marcação e o Pb sendo um valor atualizado de maneira linear começado em 0 chegando até *maxp*, sendo este último, composto pela subtração de *minth* do *avg* posterior dividindo pelo resultado da operação *maxth* menos o *minth,* assim sendo agravada lentamente (ABOUZEID; ROY, 2000; FENG *et al,* 2002).

O RED possui princípios semelhantes ao SFB, porém seus modelos de tratamento de pacotes são diferentes, como será apresentado na seção 3.3.

3.3 *Stochastic Fair Blue* (SFB)

O *Stochastic Fair Blue* (SFB) é um algoritmo de gerenciamento ativo de fila (AQM) que possui como objetivo a administração de congestionamentos agindo com base nos registros de uso da capacidade do enlace e também nas métricas de perdas de pacotes para determinar a probabilidade para marcação de fluxos. Desta forma quando tais parâmetros variam conforme a situação da rede, tal taxa será automaticamente ajustada para se adequar a demanda (FENG *et al,* 2002).Seu desenvolvimento foi baseado no algoritmo BLUE, o qual visa um tratamento justo para o maior número de fluxos possível por meio de seu método de escalonamento. Utiliza um único valor para definir a chance de marcação dos pacotes que chegam a fila. Este número é ampliado após o limite determinado para o enfileiramento ser ultrapassado (FENG *et al,* 2002). Junto a este é adotado o *Bloom Filter* que é uma estrutura de armazenagem de dados que utiliza algoritmos de *hash* para efetuar operações matemáticas sobre os elementos de entrada comprimindo-os para gerar uma identificação individual. Desta maneira a chance de colisão irá variar conforme o algoritmo usado, o qual será responsável também por mapear um determinado elemento nas posições existentes em uma matriz binária, de forma que um valor possa ser comparado a fim de validar se o mesmo corresponde ao conteúdo, usando para tais ações uma pequena quantidade de informações (MEERSMAN; DILLON; HERRERO, 2011).

Figura 7: Algoritmo de fila BLUE



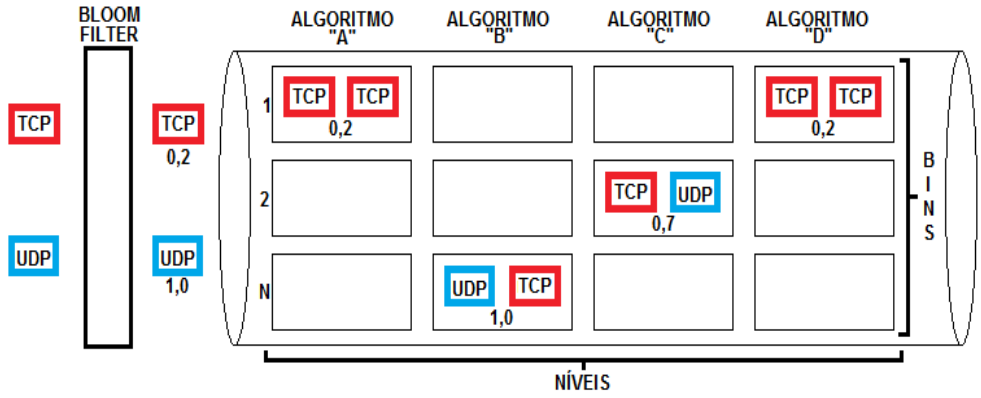
Fonte: Baseado em FENG *et al* (1999, p. 5)

Já o mecanismo de gestão de filas BLUE, mostrado na Figura 7, apresenta um modelo para atualização da taxa de probabilidade de marcação, usando para tal os parâmetros d1, d2, Pm e *freeze_time*. O d1 armazena o valor que será somado a taxa de pacotes marcados (Pm). Já o d2 mantém o valor a ser decrescido caso a fila ou enlace estejam ociosos. O *freeze_time* representa o intervalo entre duas atualizações de taxa, o qual visa evitar que uma taxa seja rapidamente alterada sem que haja tempo hábil para sua implementação e ação. Caso ocorra descarte de pacotes, ou ociosidade do enlace, será consultado o valor do parâmetro *freeze_time* antes de iniciar uma nova atualização. Se houver a confirmação que o intervalo entre as atualizações foi superado, o ajuste do parâmetro d1 ou d2 será efetivado. Para a ocorrência de perda de pacotes é somado o valor Pm ao d1. Caso o evento seja a ociosidade da posição na fila será subtraído do parâmetro d2 o valor Pm. O SFB utiliza este método para controlar a ocupação de cada posição de armazenamento, de forma que caso o espaço alocado a este se esgote, a taxa de marcação será agravada ou caso contrário a mesma será reduzida (FENG *et al,* 2002; VERMA; KUMAR, 2012).

Este modelo visa proteger o desempenho de fluxos *Transmission Control Protocol* (TCP) quando deparado com uma alta quantidade de pacotes que não utilizam regras para controle de congestionamento. Por não receber tratamento específico a chance de descarte dos pacotes TCP seria ampliada, assim gerando a retransmissão destes dados e além disso iria desperdiçar os recursos usados durante sua propagação, pois não chegará ao destino. A distinção entre estes fluxos é feita pelo *Bloom Filter* que irá avaliar o cabeçalho IP do pacote e posteriormente comparar tais informações para determinar o tipo de fluxo. Caso apurado que trata-se de fluxo não sensível a marcação, ou seja, fluxo UDP, será aplicado a ele o valor 1 referente a probabilidade de descarte, sendo possível limitar sua transmissão com a redução da largura banda disponível para este fluxo (FENG *et al,* 2002; RAMAKRISHNAN *et al,* 2001).

O SFB trabalha com o modelo de escalonamento FIFO composto por diversas posições de armazenamento (*bins*) distribuídas igualmente entre cada um dos níveis existentes, os quais possuem algoritmos de *hash* para realizar o mapeamento individual de cada fluxo e alocá-lo em uma determinada *bin*.

Figura 8: Funcionamento SFB



Fonte: Baseado em FENG *et al* (1999, p. 16)

Como cada um desses espaços podem receber vários pacotes, existe a chance de que um fluxo TCP divida este com um não sensível a marcação, assim mantendo sua chance de descarte em 1, porém ao mudar para o próximo nível passará por avaliação de seu algoritmo de *hash* e possivelmente estes fluxos serão alocados separadamente, reduzindo a probabilidade de um fluxo TCP ser confundido e indevidamente tratado, situação análoga a Figura 8. Caso a quantidade de pacotes de fluxos não sensíveis às marcas exceda a de posições de armazenamento uma menor quantidade de níveis será mais indicado para tratá-los, já que cada um desses fluxos irá poluir uma *bin* por nível e se a situação for oposta a essa, quanto maior seja a quantidade de níveis menor será a possibilidade de um fluxo TCP ser indevidamente tratado (FENG *et al,* 1999; FENG *et al,* 2002; VERMA; KUMAR, 2012).

A marcação ocorre quando um pacote entra na fila, permitindo assim que ele seja alocado em uma determinada posição de armazenamento em cada um dos níveis existentes, desta maneira gerando individualmente um registro baseado nas informações dos endereços e portas de origem e destino de cada pacote, podendo apurar o índice de ocupação referente a cada posição de armazenamento, sendo possível que uma taxa individual e eficiente de marcação seja estabelecida. Além disso a etapa de marcação tentará ligar os dois bits ECN, os quais servirão para notificar o destinatário sobre o congestionamento existente, de forma que ele solicite ao remetente que reduza a taxa de envio para evitar o descarte de pacotes, assim caso o ECN de um pacote não possa ser usado, geralmente pertencentes a

aplicações de dados contínuos, este será automaticamente descartado (FENG *et al,* 1999; FENG *et al,* 2002; RAMAKRISHNAN *et al,* 2001).

A implementação do algoritmo SFB nos sistemas operacionais baseados no GNU/Linux permite o uso de até 128 posições de armazenamento divididas entre 8 níveis com 16 bins cada. Também existem opções para configurar a forma de tratamento dos fluxos pelo SFB (TCSFB, 2011). Um cenário para demonstrar o desempenho do *DiffServ* em conjunto com a disciplinas de fila RED ou SFB será apresentado no Capítulo 4.

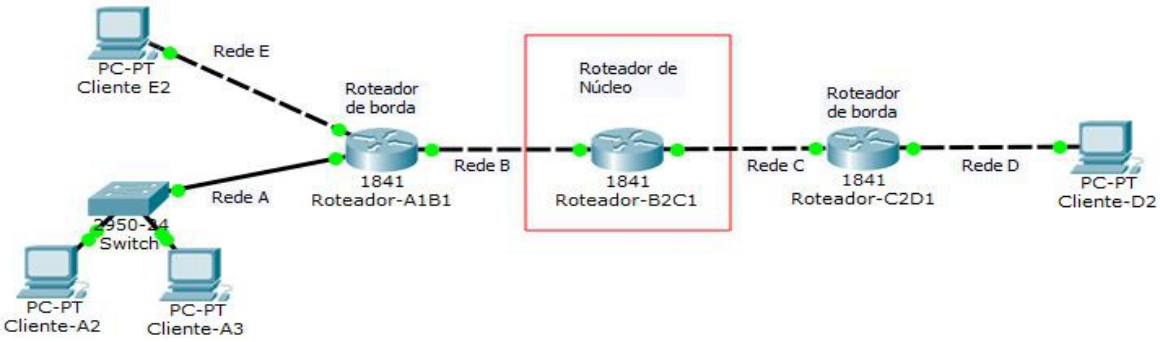
4. ESTUDO DE CASO

A proposta do experimento é demonstrar o desempenho de fluxos contínuos e não contínuos em uma rede com a arquitetura DiffServ, a qual visa prover tratamento justo para todos, a fim de não comprometer, por exemplo, o tráfego decorrente de acessos a páginas de Internet por conta do consumo de *streaming* ou vice versa.

4.1 Ambiente de Testes

Foram usadas sete máquinas virtualizadas para montagem do ambiente, as quais trabalhavam com sistema operacional Debian Wheezy (7.7) e foram divididas em cinco redes IPv6 distintas, sendo endereçadas como 2001:80:120:a::/64, 2001:80:120:b::/64, 2001:80:120:c::/64, 2001:80:120:d::/64 e 2001:80:120:e::/64 as quais serão referenciadas respectivamente por A, B, C, D e E, conforme o cenário apresentado na Figura 9.

Figura 9: Rede virtualizada dos testes



Fonte: Autoria Própria adaptado do software Packet Tracer[3].

Foram utilizadas sete máquinas virtuais na composição do laboratório de testes, as quais foram dividas entre roteadores e clientes/servidores. Três atuaram como roteadores e contavam com 768MB de memória RAM cada. O roteador de borda A1B1 ficou responsável por interligar as redes A, B e E e realizar o controle de admissão de fluxos, controlando a entrada de fluxos na rede. Os fluxos admitidos recebiam marcação, feita pela aplicação Ip6tables[4], no campo DSCP do cabeçalho

3 Informações disponíveis no sítio https://www.netacad.com/about-networking-academy/packet-tracer

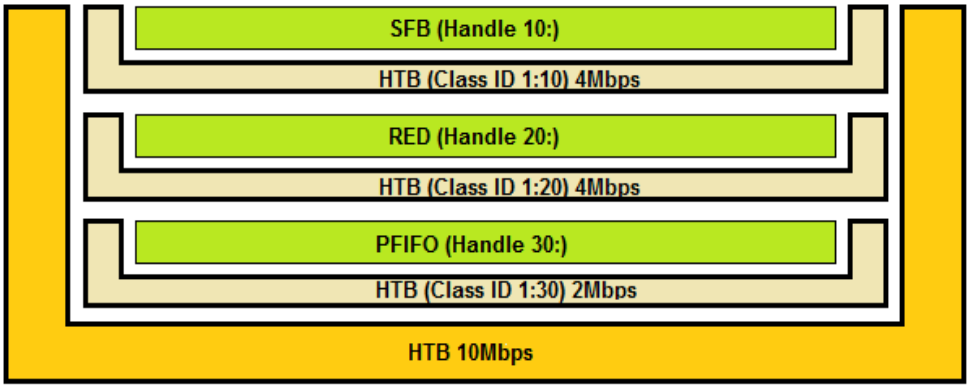
4 Informações disponíveis no sítio http://www.netfilter.org/projects/iptables/index.html

IPv6, de forma que houvesse o tratamento de fluxos TCP e UDP. Tal marcação visava identificar e diferenciar os pacotes pertencentes a uma aplicação ou determinado endereçamento, a fim de oferecer-lhes tratamento preferencial nos próximos nós conforme sua prioridade. O fluxo marcado era encaminhado para o roteador de núcleo. O roteador B2C1ao receber os pacotes enfileirava-os de acordo com sua marcação e então transmitia-os para o enlace seguinte. Assim que os pacotes eram recepcionados pelo roteador de borda C2D1 eram entregues ao endereço destinado da rede D.

As quatro máquinas restantes possuíam 512MB de RAM e uma interface de rede cabeada cada, sendo que o cliente A2, A3 e E2, endereçados respectivamente com os IPs 2001:80:120:a::2, 2001:80:120:a::3 e 2001:80:120:e::2, enviavam pacotes TCP e UDP destinados ao cliente D2, 2001:80:120:d::2, com o aplicativo Iperf[5].

O cenário montado contava com um enlace de 100 Megabit por segundo (Mbps) conectando os roteadores A1B1 e B2C1, sendo que o enlace seguinte que era destinado ao nó C2D1 dispunha de 10 Mbps. Desta maneira gerando um gargalo na rede que ocasionaria congestionamento caso o roteador de núcleo não estivesse apto a tratá-lo.

Figura 10: Estrutura das Disciplinas de Filas dos Testes



Fonte: Autoria Própria

Para tratar o congestionamento foram adotados os mecanismos de escalonamento *Packet First In First Out* (PFIFO), RED e SFB, limitados a 10 Mbps de banda, conforme apresentado na Figura 10. A disciplina PFIFO foi usada para representar o desempenho de melhor esforço (BE), padrão da internet atual, para

5 Informações disponíveis no sítio http://www.iperf.fr

tratamento de fluxos mistos, TCP e UDP. Os algoritmos de gerenciamento ativo de fila RED e SFB tiveram sua eficácia analisada dentro da arquitetura *DiffServ*, por meio do processamento de fluxos TCP e UDP com marcação DSCP.

A disciplina de fila *Hierarchical Token Bucket* (HTB) é um escalonador *Classful*, ou seja, permite criar diversas filas internas para atender diversos níveis de prioridade. Desta maneira utilizando o HTB foi possível criar uma estrutura pai, com a finalidade de limitar o total de banda disponibilizado. O HTB também foi usado para criar uma estrutura filho que limitava a banda mínima e máxima disponível para as disciplinas de fila PFIFO, RED e SFB. Tais mecanismos escalonadores são consideradas *Classless*, ou seja, não permite a criação de filas internas tratando os pacotes de forma similar.

4.2 Testes Realizados

Foram realizados oito tipos de testes a fim de avaliar o desempenho da rede sob as mais variadas situações, podendo desta forma registrar os resultados para posterior comparação.

O primeiro tipo de teste visava analisar o desempenho do mecanismo de escalonamento PFIFO, simulando o modelo *Best Effort* (BE), sobre dois fluxos UDP e um TCP. O modelo BE oferece tratamento igualitário a todos os dados, de forma que neste cenário tanto o acesso a página web quanto o *streaming* ou Videoconferência disputariam a mesma banda e sofreriam com perda de pacotes e atrasos existentes.

Já no segundo teste, a arquitetura BE recebeu dois fluxos TCP e um UDP os quais foram tratados sem diferenciação, respeitando o conceito de melhor esforço. Assim todos os pacotes eram enfileirados e transmitidos conforme sua ordem de chegada, de forma que o congestionamento iria ocasionar no acionamento do mecanismo *Slow Start* do protocolo TCP, que iria diminuir a taxa de envio após detectar a perda de pacote. A partir do terceiro até o sexto teste foi configurada a arquitetura *Diffserv* na rede, visando oferecer serviços adicionais para tratamento dos fluxos, conforme apresentado na Seção 3.1.

Para o terceiro teste foi adotado o gerenciador ativo de filas RED para tratamento de fluxos mistos, compostos por dois fluxos UDP e um fluxo TCP. O RED

foi avaliado com relação a seu desempenho perante um congestionamento decorrente do gargalo existente no enlace que conecta as redes C e D.

O tráfego TCP simbolizava acesso de página web e o tráfego UDP representava áudio streaming por máquinas da rede A e E. Todos os pacotes de aplicações de áudio receberam a marcação AF13 em seu campo DSCP e o fluxo *Hypertext Transfer Protocol* (HTTP) a marca EF, conforme Tabela 1.

Tabela 1: Identificação marcação DSCP / DS

Classe	DSCP		DS	
	Binário	Hexadecimal	Binário	Hexadecimal
AF11	001010	0xa	00101000	0x28
AF12	001100	0xc	00110000	0x30
AF13	001110	0xe	00111000	0x38
AF21	010010	0x12	01001000	0x48
AF22	010100	0x14	01010000	0x50
AF23	010110	0x16	01011000	0x58
AF31	011010	0x1a	01101000	0x68
AF32	011100	0x1c	01110000	0x70
AF33	011110	0x1e	01111000	0x78
AF41	100010	0x22	10001000	0x88
AF42	100100	0x24	10010000	0x90
AF43	100110	0x26	10011000	0x98
EF	101110	0x2e	10111000	0xb8

Fonte: Autoria Própria com base no modelo (PINTO *et al,* 2003, p. 333)

No quarto teste o RED recebeu dois fluxos TCP e um UDP. A marcação dos pacotes para definir sua prioridade de tratamento foi feita com base em suas aplicações, seguindo o mesmo conceito do Terceiro teste. Desta forma foi possível avaliar seu desempenho quando os fluxos TCP representam mais que 50% dos pacotes processados pelo roteador.

Já no quinto teste foi adotado o mecanismo de escalonamento SFB, que recebeu dois fluxos UDP simbolizando dados de Voz sobre IP (VoIP) e um fluxo TCP representado uma transferência *File Transfer Protocol* (FTP). Neste caso a marcação ocorreu baseada no endereço de rede dos pacotes ao invés do tipo de aplicação. Para os pacotes TCP e UDP originados na rede A foi aplicada a marcação EF. Já o fluxo UDP gerado pela rede E foi marcado com AF11.

O sexto teste foi feito sobre a disciplina SFB recebendo dois fluxos TCP, simbolizando dados FTP, e um fluxo UDP, representando uma chamada VoIP. Sendo que a rede A recebeu a marcação EF tanto nos pacotes UDP quanto TCP e a rede E

a marca AF11. Desta forma durante o congestionamento foi possível avaliar o desempenho obtido pelo SFB ao alocar os pacotes entre as posições de armazenamento (bin) disponíveis.

O sétimo e oitavo testes foram feitos usando dois fluxos TCP e quatro UDP, expondo o cenário BE e SFB a alta quantidade de pacotes injetados, disputando a banda disponível e sobrecarregando o *buffer* de fila.

Em todos os testes os pacotes foram originados das redes A e E, sendo propagados após o roteador de borda por uma banda de 100 Mbps e posterior ao enfileiramento eram transmitidos por um enlace com 10 Mbps até o roteador de borda de seu destino. O roteador de núcleo manteve a disciplina de fila HTB pai limitando a banda para 10 Mbps, oferecendo um limite mínimo de 4 Mbps e máximo de 10 Mbps para os escalonadores HTB filhos que faziam base para as disciplinas de escalonamento RED e SFB. Sendo que o modelo BE, representado pelo mecanismo PFIFO, mantinha um mínimo de 2 Mbps podendo alcançar até 10 Mbps caso a banda estivesse ociosa. Cada fluxo TCP e UDP dos testes foi capturado individualmente permitindo que fossem apurados os atrasos sofridos, perda de pacotes e banda utilizada por cada um.

De maneira geral, os testes simularam um cenário em que havia tráfego misto transmitido entre as cinco redes conectadas. Os dados eram gerados pelas estações A2, A3 e E2 e destinados a estação D2, sendo compostos por pacotes de aplicações transportados pelo protocolo TCP, como HTTP e FTP, e pelo protocolo UDP, como *streaming* e VoIP. O ingresso ao domínio *DiffServ* era controlado pelo roteador de borda A1B1 que efetuava a admissão e marcação dos pacotes e transmitia-os pelo enlace com vazão de 100 Mbps. Ao ser recebido pelo roteador de núcleo B2C1 era enfileirado e processado sobre as regras da disciplina de escalonamento RED ou SFB, sendo posteriormente transmitido pelo enlace com vazão de 10 Mbps. O roteador de borda C2D1 ao receber o pacote fazia o direcionamento para a máquina D2. O cenário para a arquitetura BE era similar, porém não oferecia tratamento e serviços adicionais, assim admitindo todos os pacotes e os propagando pela rede conforme o modo de melhor esforço. Após a coleta de informações por meio de cada teste, foi possível estudar estes dados e mensurar seus resultados, esboçando estes graficamente conforme apresentado no Capítulo 5.

5. RESULTADOS OBTIDOS

Foram feitas três coletas de dados para cada tipo de teste apresentado na Seção 4.2, por meio de capturas com o software Tcpdump[6]. O roteador C2D1 foi o responsável por coletar as informações dos fluxos que chegavam destinados a máquina D2, permitindo assim contabilizar os pacotes que eram capturados, recebidos e os descartados. As máquinas A2, A3 e E2 usavam a aplicação Iperf para gerar tráfego TCP e UDP destinado a rede D. O envio de cada fluxo era feito por um período de 15 segundos e após o término, o Iperf apresentava as informações do fluxo sobre a banda usada, total de bytes e quantidade de pacotes transferidos.

Os dados obtidos por meio dos testes foram analisados e após apuração, foram dispostos em planilha e representados em gráficos para permitir uma apresentação objetiva de seus resultados. Desta forma foi possível efetuar o cálculo médio para pacotes recebidos, taxa de descarte e banda dedicada para cada fluxo.

Tabela 2: Testes com dois fluxos TCP e um UDP

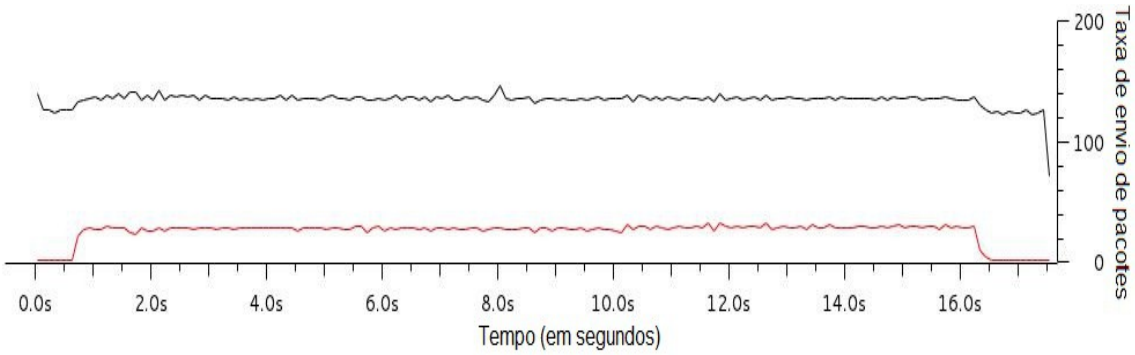
Escalonador	Média de Pacotes Recebidos	Taxa de Descarte	Média de Perdas	Banda Média por Fluxo (Mbps)		
				UDP rede A	TCP rede A	TCP rede E
SFB	18496	0,016%	3	3,21	3,81	3,58
RED	17996	0,126%	23	3,27	3,00	3,99
PFIFO	18375	0,288%	53	3,21	3,89	3,43

Fonte: Autoria Própria.

A Tabela 2 demonstra os resultados dos experimentos feitos com dois fluxos TCP e um UDP tratados pelos mecanismos de escalonamento PFIFO, RED e SFB. O desempenho do SFB apresentou-se melhor que o PFIFO e RED, tendo uma média de 0,016% de pacotes descartados. O modelo PFIFO foi testado em um cenário que não há tratamento diferenciado do tráfego, de forma que todos os fluxos foram enfileirados e processados sem receber priorização. Neste modelo a taxa contabilizada de perda de pacotes foi de 0,288%, a qual é 1800% superior ao valor obtido pelo SFB. Já a disciplina de fila RED obteve uma taxa média de descarte de 0,126%, que é menor que a metade do valor apresentado ao usar a arquitetura padrão de melhor esforço (BE).

6 Informações disponíveis no sítio http://www.tcpdump.org

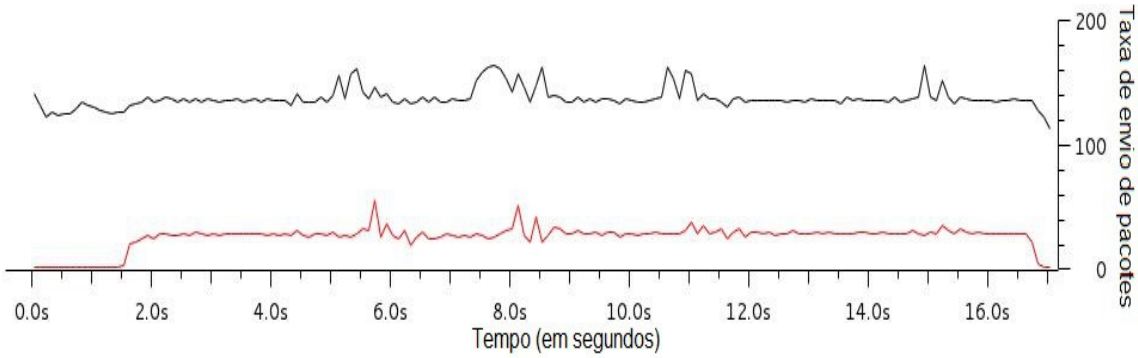
Figura 11: Desempenho SFB com dois fluxos TCP e um UDP



Fonte: Autoria Própria.

Na Figura 11 é representado graficamente o desempenho do mecanismo de escalonamento SFB, o qual se manteve linear e com suaves oscilações na taxa de transmissão de pacotes. A linha em preto se refere aos fluxos TCP e a em vermelho ao tráfego UDP, as quais demonstram uma gestão de congestionamento eficiente para tráfegos mistos TCP e UDP, com maior concentração de pacotes TCP.

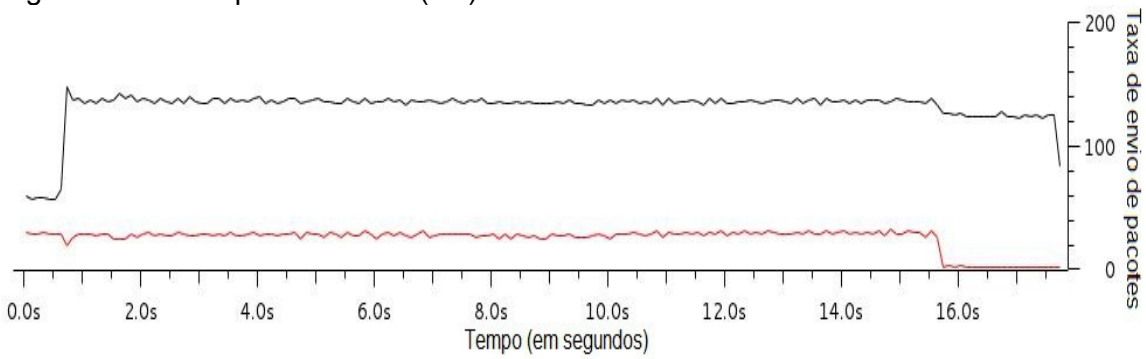
Figura 12: Desempenho RED com dois fluxos TCP e um UDP



Fonte: Autoria Própria.

Já o escalonador RED sofreu maiores oscilações tanto na propagação dos tráfegos TCP, linha de cor preta, e fluxo UDP, linha de cor vermelha, conforme apresentado na Figura 12. Desta forma o RED não manteve a uniformidade nas taxas de envio de pacotes, porém manteve a efetividade no controle de congestionamento, minimizando os impactos sobre os fluxos. As oscilações nas taxas de transmissão são apresentadas no gráfico nos intervalos entre 5 a 6,5, 7 a 8,5, 10,5 a 11,5 e 14,5 a 15,5 segundos.

Figura 13: Desempenho PFIFO (BE) com dois fluxos TCP e um UDP



Fonte: Autoria Própria.

Os fluxos TCP e UDP tratados pelo modelo BE, exibidos na Figura 13, foram transmitidos de forma uniforme, porém por não contar com mecanismos para controle de congestionamento realizou vários descartes.

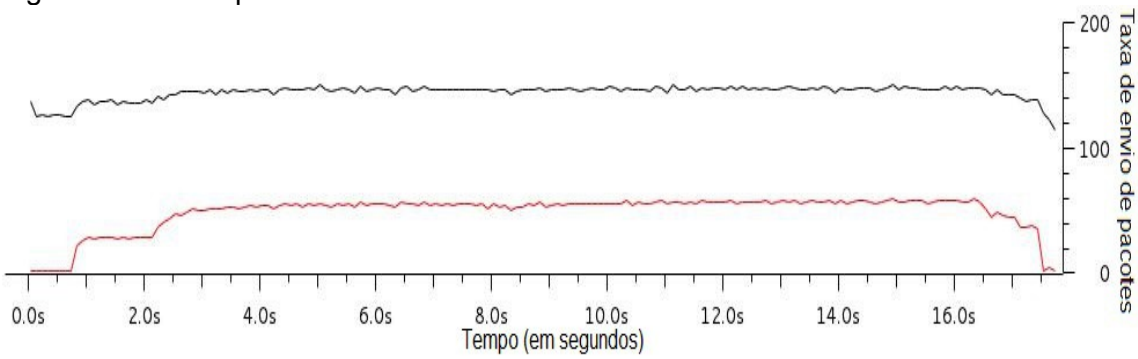
Tabela 3: Testes com dois fluxos UDP e um TCP

Escalonador	Média de Pacotes Recebidos	Taxa de Descarte	Média de Perdas	Banda Média por Fluxo (Mbps)		
				TCP rede A	UDP rede A	UDP rede E
SFB	21927	0,166%	36	3,66	3,28	3,19
RED	21768	0,181%	39	3,58	3,28	3,29
PFIFO	22136	0,532%	118	3,77	3,20	3,24

Fonte: Autoria Própria.

Os testes realizados com dois fluxos UDP e um TCP apresentaram maior volume de descarte de pacotes, sendo justificável pelo fato dos pacotes UDP não acatarem a marcação ECN. Desta forma pelo maior volume de fluxo UDP propagado na rede, os mecanismos de escalonamento RED e SFB controlaram melhor o congestionamento que o modelo PFIFO. O representante do modelo *Best Effort* (BE) alcançou o valor de 0,532% na taxa de descarte de pacotes, o qual é mais que o dobro do valor obtido pelas disciplinas de escalonamento RED e SFB. Neste tipo de teste as disciplinas RED e SFB demonstraram resultados parecidos. O RED descartou 39 pacotes dos 21.768 enviados e o SFB descartou 36 dos 21.927 pacotes recebidos, apresentando uma taxa média de perda de 0,181% e 0,166% respectivamente, conforme apresentado na Tabela 3. A banda média para cada fluxo se mantiveram próximas, de forma que o fluxo não seria degradado por ter baixa vazão (*throughput*).

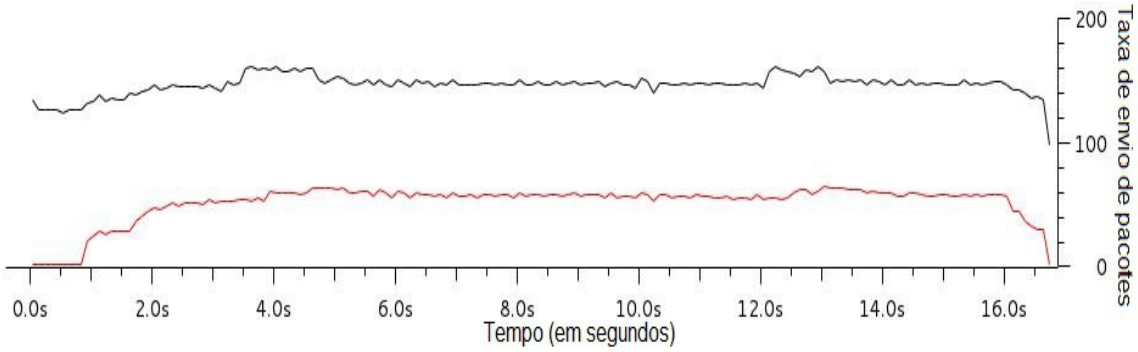
Figura 14: Desempenho SFB com dois fluxos UDP e um TCP



Fonte: Autoria Própria.

Conforme exposto na Figura 14, o SFB manteve uma taxa uniforme de envio de pacotes TCP, linha preta, mesmo com a injeção de dois fluxos UDP, linha vermelha. O SFB aumentou a taxa de descarte para controlar o congestionamento, já que a maior parte dos pacotes não são sensíveis a marcação ECN. Mesmo com maior índice de perda de pacotes o SFB se mostrou eficaz tanto para proteção do fluxo TCP quanto na mitigação de congestão.

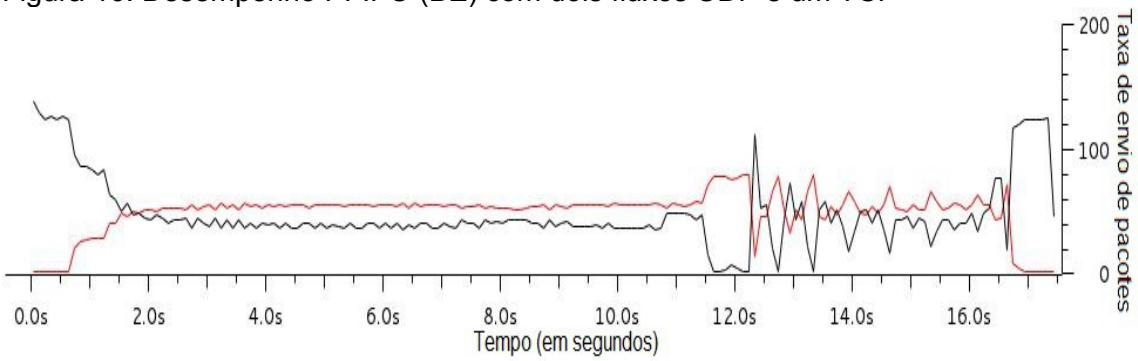
Figura 15: Desempenho RED com dois fluxos UDP e um TCP



Fonte: Autoria Própria.

O RED apresentou um desempenho próximo ao SFB em relação ao controle de descarte, obtendo a taxa de 9,1% acima do que a obtida pelo SFB nos testes realizados com dois fluxos UDP, representados pela linha inferior, e apenas um TCP, representado na pela linha superior. Na Figura 15 é demonstrado a quantidade de pacotes enviados durante o período do teste, sofrendo oscilações leves em sua taxa de transmissão, conforme pode ser visto nos intervalos entre os segundos 3,5 a 5 e 12 a 13.

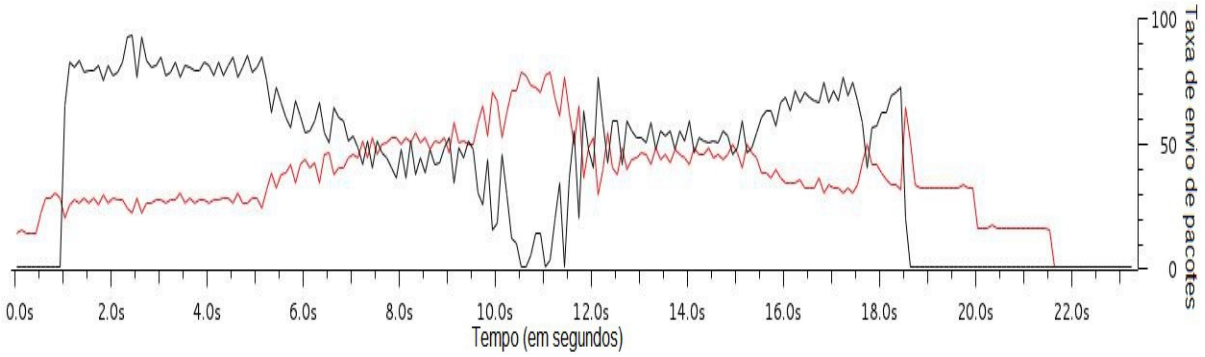
Figura 16: Desempenho PFIFO (BE) com dois fluxos UDP e um TCP



Fonte: Autoria Própria.

Com os dados coletados no experimento com dois fluxos UDP e um TCP sobre desempenho do PFIFO, foi possível as representações gráficas da Figura 16, que demonstram a degradação de cada tráfego. Após algum tempo de envio de pacotes ocasionou congestionamento mediante ao elevado número de pacotes e o gargalo existente no enlace entre as redes C e D. A linha em preto se refere ao tráfego de pacotes TCP e a em vermelho aos fluxos UDP transmitidos.

Figura 17: Desempenho PFIFO (BE) com quatro fluxos UDP e dois TCP



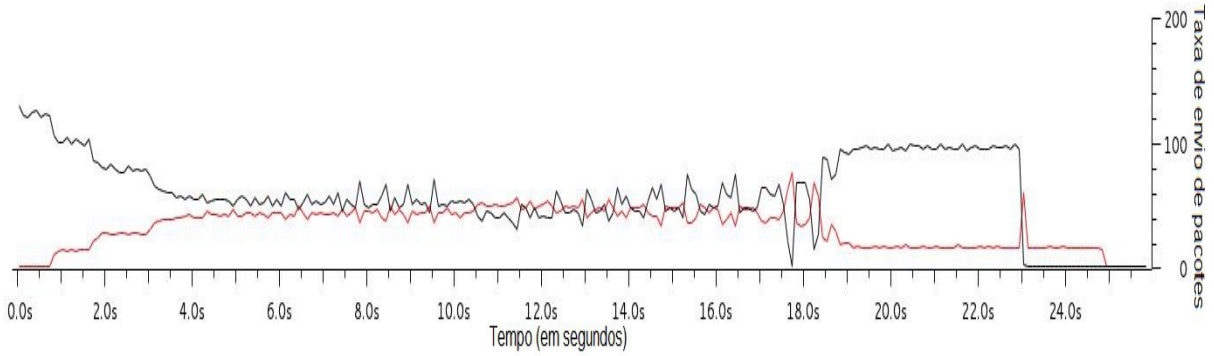
Fonte: Autoria Própria.

A Figura 17 demonstra o desempenho do BE ao receber seis fluxos distintos, dos quais dois são TCP e os outros quatro UDP, de forma que não existe proteção ao TCP. Assim a taxa dos fluxos TCP é denegrida ao disputar banda os fluxos UDP.

Já a Figura 18 apresenta o desempenho da disciplina de fila SFB, que mesmo recebendo uma elevada quantidade de pacotes UDP, conseguiu manter a banda média dos fluxos TCP entre 4,10 Mbps, a qual teve um valor mínimo de 2,29 Mbps e máximo de 5,15 Mbps. Os fluxos UDP alcançaram taxas inferiores a 1 Mbps e desta forma mesmo em uma condição de congestionamento protegeu os pacotes TCP.

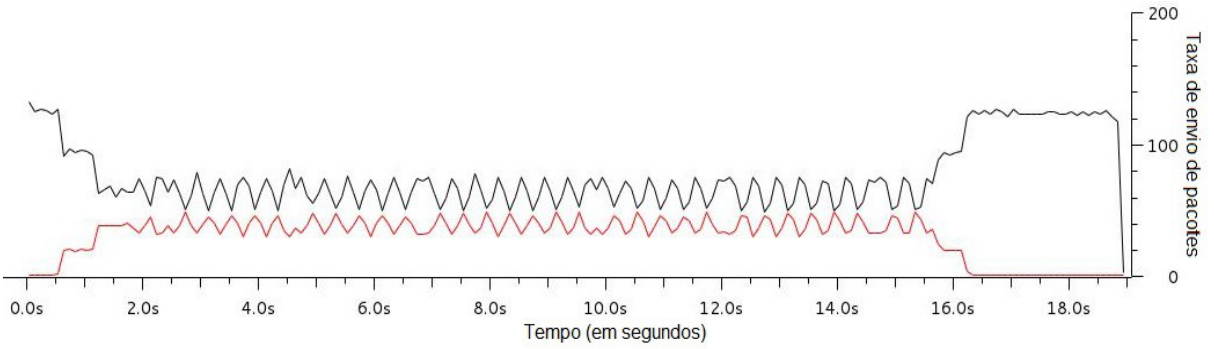
Desta forma o SFB atendeu plenamente seu objetivo, assegurando altas taxas de transmissão ao fluxo TCP mesmo em situações extremas. A eficiência do mecanismo para controle de congestionamento do SFB foi comprovada, atuando conforme sua proposta.

Figura 18: Desempenho SFB com quatro fluxos UDP e dois TCP



Fonte: Autoria Própria.

Figura 19: Desempenho SFB com tráfegos TCP e UDP com QoS



Fonte: Autoria Própria.

Utilizando o SFB em tráfegos TCP e UDP com marcações distintas, conforme Figura 19, assegura a propagação quase uniforme dos fluxos, sem que haja disputa por banda e consequentemente a degradação de determinado fluxo. No exemplo foi usada QoS para limitar a taxa de transferência de cada um dos dois fluxos UDP a 2,5 Mbps. Assim os dois fluxos TCP alcançaram a taxa média de 3,55 Mbps, por conta da proteção provida por meio dos mecanismos de controle de congestionamento da disciplina de fila SFB em conjunto com o controle da banda provida aos tráfegos.

6. CONSIDERAÇÕES FINAIS

Durante a elaboração deste trabalho foram abordados temas para fundamentar o uso de uma rede IPv6 com QoS, visando oferecer tratamento justo para todo o tráfego gerado por aplicações que utilizam os protocolos TCP ou UDP em sua camada de transporte. Desta forma a propagação de fluxos mistos pela rede ocorre sem que haja elevada degradação da banda disponível ou agravo na taxa de descarte, mesmo em período de congestionamento. A utilização da sexta versão do protocolo IP (IPv6) pôde demonstrar que a estruturação da rede é possível e que os resultados apresentados por este documento serão válidos com este recente modelo do protocolo IP.

Foi apresentada a definição de qualidade de serviço (QoS) e os parâmetros existentes em uma rede, como atraso de enfileiramento, vazão e perda de dados, os quais foram focados pelo trabalho. Visando o controle destes parâmetros foi proposta a montagem do cenário sobre uma arquitetura *Differentiated Services* (DiffServ), de forma que qualquer pacote pudesse ser marcado para receber tratamento preferencial. Desta maneira durante o período de admissão do fluxo foi possível identificar informações como endereço de origem, destino e de rede, protocolo usado ou outros dados contidos no cabeçalho IP, aplicando sobre cada fluxo individualmente uma marca entre para sua identificação pelos próximos nós. Após a marcação os pacotes foram ingressados por um enlace de 100 Mbps e ao serem recebidos pelo roteador de núcleo (RN), eram enfileirados conforme seu nível de priorização. Nesta etapa os dados eram tratados pelas disciplinas de fila RED ou SFB, ambos com mecanismos para mitigação dos efeitos do congestionamento, que poderiam elevar a taxa de descarte de pacotes por sobrecarga do *buffer* do roteador. Posteriormente os pacotes eram transmitidos por um enlace de 10 Mbps, o qual por ser inferior ao enlace de entrada do RN ocasionava atraso de enfileiramento e aumentava a probabilidade de um congestionamento ocorrer.

Para comparar o desempenho dos gerenciadores ativo de fila RED e SFB, os quais utilizam métodos para controle de congestionamento e mitigação da taxa de descarte, com o modelo padrão usado na Internet, o *Best Effort* (BE) ou melhor esforço, o cenário do teste foi mantido, porém com a remoção do controle de admissão e com o uso da disciplina de fila PFIFO, de forma que nenhum fluxo seria priorizado.

Analisando o resultado dos testes, foi possível comparar o desempenho da propagação de pacotes em uma rede com QoS com uma rede sem QoS. Assim foi apurado que QoS em uma rede que atua com dados mistos, garante um melhor desempenho para todas as aplicações, de modo que a implementação de QoS permite administrar melhor os recursos, a fim de atender adequadamente a necessidade de cada fluxo de dados. Baseando-se nos resultados obtidos durante todo o projeto mesmo com o crescimento do uso da redes pelos mais variados dispositivos, quanto melhor estruturada a rede em que estes serão propagados, mais satisfatória será a qualidade de experiência (QoE) percebida pelos usuários.

Também foi possível apurar que tanto o RED quanto o SFB possuem desempenho superior ao PFIFO, que representou o modelo de melhor esforço (BE). Conforme os tipos de experimentos realizados o mecanismo de escalonamento SFB apresentou-se superior, garantindo que o tráfego TCP não sofresse grandes impactos ao disputar recursos com fluxos UDP. Já a disciplina de fila RED possui bom desempenho porém não oferece proteção ao fluxo TCP, de modo que caso o mesmo seja descartado ou marcado no campo ECN, ativará o mecanismo de *Slow Start* do protocolo TCP, de forma que haverá maior chance de ocorrerem congestionamentos. Desta forma o RED obteve desempenho inferior ao SFB, porém amplamente superior ao PFIFO. O PFIFO representou bem o modelo BE, de forma que a ordem de chegada do pacote seria sua ordem de transmissão e ao haver esgotamento do *buffer* de fila, todos os pacotes posteriores eram descartados.

Este trabalho demonstrou os benefícios do uso de QoS para tratamento de fluxos mistos, tanto fluxos TCP quanto fluxos UDP, porém seu estudo pode ser continuado, visando aperfeiçoar o desempenho deste modelo de rede. Sugere-se a elaboração de um cenário que disponha de mais roteadores de núcleo e que utilize mais de uma disciplina de fila, montando desta maneira uma hierarquia na qual a priorização do pacote, por meio de sua marcação, em conjunto com determinado mecanismo de escalonamento atenda adequadamente a diversos tipos de fluxos simultaneamente.

REFERÊNCIAS

ABOUZEID, A.; ROY, S. In: IEEE Global Communications Conference, San Francisco. **Modeling Random Early Detection in a Differentiated Services Network**. 2000.

ALLMAN, M. et al. **TCP Congestion Control**. 2009. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc5681.txt>. Acesso em: 11 fev 2015.

BLAKE, S. et al. **An Architecture for Differentiated Services**. 1998. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2475.txt>. Acesso em: 08 nov 2013.

BRADEN, R.; CLARK, D.; SHENKER, S. **Integrated Services in the Internet Architecture:** An overview. 1994. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc1633.txt>. Acesso em: 21 ago 2013.

BRADEN, E. R. et al. **Resource Reservation Protocol (RSVP)**. 1997. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2205.txt>. Acesso em: 18 jun 2013.

Cisco Systems. **Enterprise QoS solution reference network design guide**. 2005. San Jose. Disponível em: <http://www.cisco.com/en/US/docs/solutions/Enterprise/WAN_and_MAN/QoS_SRND/QoS-SRND-Book.html>. Acesso em: 25 out 2013.

DALBEM, E. **Qualidade de Serviço para Aplicações de Rede sobre IPv6**. 2014. 74 folhas. Trabalho Científico – Curso Superior de Tecnologia em Segurança da informação, Faculdade de Tecnologia de Americana.

DAS, K. **IPv6 - The History and Timeline**. 2008. IPv6.com Inc. Disponível em: <http://www.ipv6.com/articles/general/timeline-of-ipv6.htm>. Acesso em: 14 abr 2015.

DAVIE, B. et al. **An expedited forwarding PHB (Per-hop Behavior)**. 2002. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc3246.txt>. Acesso em: 17 jun 2013.

DEERING, S. **Internet Protocol, version 6 (IPv6)**. 1998. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2460.txt>. Acesso em: 17 jun 2013.

DEGERMARK, M. et al. **IP Header Compression**. 1999. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2507.txt>. Acesso em: 14 abr 2015.

FENG, W. et al. In: CSE-TR-387-99, Michigan. **BLUE:** A new class of active queue management algorithms. University of Michigan, 1999.

FENG, W. et al. **The BLUE Active Queue Management Algorithms.** IEEE/ACM Transactions on Networking, Vol. 10, Número 4. Páginas 513-528. Agosto 2002.

FLOYD, S.; JACOBSON, V. In: IEEE/ACM Transactions on Networking. **Random Early Detection Gateways for Congestion Avoidance**. 1993. páginas 397 – 413.

FLOYD, S.; ALLMAN, M. **Comments on the Usefulness of Simple Best-Effort Traffic**. 2008. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc5290.txt>. Acesso em: 20 fev 2015.

FOROUZAN, B. **Comunicação de Dados e Redes de Computação**. 4ª edição. Porto Alegre: AMGH editora, 2008. 1103 páginas.

GROSSMAN, D. **New terminology and clarifications for DiffServ**. 2002. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc3260.txt>. Acesso em: 21 jun 2013.

GUNTER, M.; BRAUN, T. In: Internacional Conference On Networks Protocols, 7º, ICNP 1999.801928. **Evaluation of Bandwidth Broker Signaling**. 1999. páginas 145 – 152.

HEINANEN, J. **Assured forwarding PHB group**. 1999. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2597.txt>. Acesso em: 17 jun 2013.

JACOBSON, V. et al. **An expedited forwarding PHB**. 1999. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2598.txt>. Acesso em: 17 jun 2013.

KUROSE, J.; ROSS, K. **Redes de Computadores e a Internet:** Uma abordagem top-down. 5ª edição. São Paulo: Pearson Education, 2010. 614 páginas.

MEERSMAN, R.; DILLON, T.; HERRERO, P. **On The Move To Meaningful Internet Systems:** OTM 2011 workshops. Editora: Springer. Grécia, 2011. Página 537.

NORDMARK, E.; GILLIGAN, R. **Basic transition mechanisms for IPv6 hosts and routers**. 2005. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc4213.txt>. Acesso em: 19 jun 2013.

NICHOLS, K. et al. **Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers**. 1998. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2474.txt>. Acesso em: 24 jun 2013.

PATRICK, A. et al. In**:** International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks, 1º, Dallas. **A QoE Sensitive Architecture for Advanced Collaborative Environments**. 2004.

PINTO, R. et al. In**:** Simpósio Brasileiro de Redes de Computadores, 21º, Natal. **Incorporação de qualidade de serviços em aplicações telemáticas**. Natal: SBC, 2003. páginas 331 – 346.

RAMAKRISHNAN, K. et al. **The Addition of Explicit Congestion Notification (ECN) to IP**. 2001. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc3168.txt>. Acesso em: 10 mar 2015.

ROSEN, E. et al. **Multiprotocol Label Switching architecture**. 2001. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc3031.txt>. Acesso em: 24 nov 2013.

SANTOS, R. et al. **Apostila "Curso IPv6 básico"**. Núcleo de Informação e Coordenação do ponto BR. 2011. São Paulo. Disponível em: <http://curso.ipv6.br>. Acesso em: 18 out 2013.

SHENKER, S. et al. **Specification of Guaranteed Quality of Service**. 1997. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2212.txt>. Acesso em: 18 out 2013.

STALLINGS, W. **Redes e Sistema de Comunicação de Dados**. 5ª edição. Rio de Janeiro: Elsevier, 2005. 439 páginas.

STEVENS, W. **TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms.** 1997.Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2001.txt>. Acesso em: 13 mar 2015.

TANENBAUM, A. **Redes de Computadores**. 4ª edição. Rio de Janeiro: Elsevier, 2003. 920 páginas.

TC Qdisc: SFB - Stochastic Fair Blue**.** Linux Man Pages. 2011. Disponível em: <http://man7.org/linux/man-pages/man8/tc-sfb.8.html>. Acesso em: 14 mar 2015.

VERMA, H.; KUMAR, A. **Performance Evaluation of AQM Techniques in PIM-DM Multicast Network for SRM Protocol.** Intenational Journal of Computer Applications, 48º. Páginas 6-10. Junho 2012.

WROCLAWSKI, J. **The Use of RSVP with IETF Integrated Services.** 1997.Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2210.txt>. Acesso em: 29 jan 2014.

WROCLAWSKI, J. **Specification of the Controlled-Load Network Element Service**. 1997a. Internet Engineering Task Force. Disponível em: <http://www.ietf.org/rfc/rfc2211.txt>. Acesso em: 17 jun 2013.

ZHENG, B.; ATIQUZZAMAN, M. In**:** International Journal of Communication System. 15º. **Low pass filter/over drop avoidance (LPF/ODA):** an algorithm to improve the response time of RED gateways. 2002. páginas 899 – 906.

GLOSSÁRIO

ACK: *Acknowledgement* representa a confirmação da recepção de um pacote.

AF: *Assured Forwarding* ou Encaminhamento Assegurado.

AQM: *Active Queue Management* ou gerenciamento ativo de fila.

BB: *Bandwidth Broker*, administrador da banda a ser disponibilizada para o tráfego.

BE: *Best Effort* ou Melhor Esforço, padrão adotado em redes.

Bins: Posição de armazenamento usado pela disciplina de fila SFB.

Buffer: Espaço para armazenamento.

Cwnd: *Congestion Window*, armazena a quantidade de pacotes enviados iniciando em 1 sendo dobrado após receber um ACK.

DiffServ: *Differentiated Services* ou Serviços Diferenciados, arquitetura que efetua a marcação do cabeçalho dos pacotes.

DNS: *Domain Name System*, responsável por resolver nomes ou endereços.

DSCP: *Differentiated Service Code Point*, código aplicado ao pacote para identificação dentro de uma rede com arquitetura *DiffServ*.

ECN: *Explicit Congestion Notification*, marcação para informar sobre o congestionamento existente na rede.

EF: *Expedited Forwarding* ou Encaminhamento Expresso.

FEC: *Forward Equivalent Class*, classe que determina um grupo e seu tratamento.

FIFO: *First In First Out*, regra que atende conforme ordem de chegada.

FTP: *File Transfer Protocol*, usado para transferência de arquivos.

HTB: *Hierarchical Token Bucket*, escalonador *Classful*, ou seja, permite criar diversas filas internas para atender diversos níveis de prioridade.

HTTP: *Hypertext Transfer Protocol*.

IHL: Internet *Header Length*, campo para identificar o tamanho do cabeçalho Ipv4.

IntServ: *Integrated Services* ou Serviços Integrados, arquitetura que reserva recursos junto aos nós desde a origem até o destino.

IP: Internet *Protocol*, protocolo usado na camada de rede.

IPv4: Protocolo de Internet versão 4.

IPv6: Protocolo de Internet versão 6.

Jitter: Variação de atrasos sofrido por um fluxo.

LSR: *Label Switched Routers*, roteador MPLS que efetua a troca de rótulos.

Mbps: Megabit por segundo.

MPLS: *Multi Protocol Label Switching*, arquitetura que etiqueta pacotes da rede.

OSPF: *Open Shortest Path First*, estabelece a melhor rota para o pacote seguir.

PFIFO: *Packet First In First Out*.

PHB: *Per Hop Behavior*, define o modelo de encaminhamento que será ofertado em cada nó.

QoE: *Quality of Experience* ou Qualidade de Experiência, avaliação subjetiva obtida pelo usuário após experimentação da aplicação.

QoS: Qualidade de serviço, desempenho de um fluxo sobre os parâmetros.

RB: Roteador de Borda, roteador que fica nas bordas da rede *DiffServ*.

RED: *Random Early Detection*.

RFC: *Request For Comments*, documentação sobre protocolos e serviços.

RN: Roteador de Núcleo, roteador alocados no núcleo da rede *DiffServ*.

RSVP: *Resource Reservation Protocol*, protocolo usado pela arquitetura *IntServ* para reservar os recursos junto aos nós.

Rwnd: *Receiver Window*, quantidade de dados que o destinatário poderá receber por envio.

SFB: *Stochastic Fair Blue*.

SLA: *Service Level Agreement*, acordo estabelecido entre pontos.

SLS: *Service Level Specification*, especificações sobre os valores definidos para os parâmetros.

TCA: *Traffic Conditioning Agreement*, acordo sobre como o tráfego será condicionado.

TCS: *Traffic Conditioning Specification*, regras que detalham como o tráfego será condicionado.

TCP: *Transmission Control Protocol*.

Throughput: Vazão ou Largura de Banda.

ToS: *Type of Service*, campo IPv4 para informação sobre QoS.

TTL: Time To Live, campo IPv4 que representa a quantidade máxima de saltos.

UDP: *User Datagram Protocol*.

VoIP: *Voice over* IP ou Voz sobre IP.

APÊNDICE A

```
#!/bin/bash
####################
###  Disciplina de fila  ##
####################

# Criação do HTB raiz
tc qdisc add dev eth1 root handle 1: htb default 30
tc class add dev eth1 parent 1: classid 1:1 htb rate 10mbit burst 80k

# Criação HTBs filhos
tc class add dev eth1 parent 1:1 classid  1:10 htb rate 4mbit ceil \
10mbit burst 80k
tc class add dev eth1 parent 1:1 classid  1:20 htb rate 4mbit ceil \
10mbit burst 80k
tc class add dev eth1 parent 1:1 classid  1:30 htb rate 2mbit ceil \
10mbit burst 80k prio 3

# Regras disciplinas de fila
tc qdisc add dev eth1 parent 1:10 handle 10: sfb \
limit 4000 max 2400 target 1600 penalty_rate 1440 \
penalty_burst 2280
tc qdisc add dev eth1 parent 1:20 handle 20: red \
limit 1000000 min 400000 max 600000 avpkt 1500 burst 450 \
probability 0.02 bandwidth 4000000 ecn
tc qdisc add dev eth1 parent 1:30 handle 30: pfifo

# Filtros
tc filter add dev eth1 parent 1: protocol ipv6 u32 match u16 \
0x0b80 0x0ff0 at 0 flowid 1:10
tc filter add dev eth1 parent 1: protocol ipv6 u32 match u16 \
0x0380 0x0ff0 at 0 flowid 1:20
```

APÊNDICE B

```
#!/bin/bash

#####################
##    Admissão SFB    ##
#####################

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
ip6tables -t mangle -F
ip6tables -F

# Rede A
ip6tables -t mangle -A FORWARD -s 2001:80:120:a::/64 -p tcp -j DSCP \
--set-dscp-class EF
ip6tables -t mangle -A FORWARD -s 2001:80:120:a::/64 -p udp -j DSCP \
--set-dscp-class EF
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:a::/64 -p tcp -j DSCP \
--set-dscp-class EF
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:a::/64 -p udp -j DSCP \
--set-dscp-class EF

# Rede E
ip6tables -t mangle -A FORWARD -s 2001:80:120:e::/64 -p tcp -j DSCP \
--set-dscp-class EF
ip6tables -t mangle -A FORWARD -s 2001:80:120:e::/64 -p udp -j DSCP \
--set-dscp-class EF
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:e::/64 -p tcp -j DSCP \
--set-dscp-class EF
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:e::/64 -p udp -j DSCP \
--set-dscp-class EF
```

APÊNDICE C

```
#!/bin/bash

####################
##   Admissão RED   ##
####################

echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
ip6tables -t mangle -F
ip6tables -F

# Rede A
ip6tables -t mangle -A FORWARD -s 2001:80:120:a::/64 -p tcp -j DSCP \
--set-dscp-class AF13
ip6tables -t mangle -A FORWARD -s 2001:80:120:a::/64 -p udp -j DSCP \
--set-dscp-class AF13
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:a::/64 -p tcp -j DSCP \
--set-dscp-class AF13
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:a::/64 -p udp -j DSCP \
--set-dscp-class AF13

# Rede E
ip6tables -t mangle -A FORWARD -s 2001:80:120:e::/64 -p tcp -j DSCP \
--set-dscp-class AF13
ip6tables -t mangle -A FORWARD -s 2001:80:120:e::/64 -p udp -j DSCP \
--set-dscp-class AF13
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:e::/64 -p tcp -j DSCP \
--set-dscp-class AF13
ip6tables -t mangle -A POSTROUTING -s 2001:80:120:e::/64 -p udp -j DSCP \
--set-dscp-class AF13
```