UNIVERSIDADE FEDERAL FLUMINENSE
ESCOLA DE ENGENHARIA
MESTRADO EM ENGENHARIA DE TELECOMUNICAÇÕES

BRUNO LIMA WANDERLEY

TC MESH: UMA FERRAMENTA DE GERÊNCIA DE QUALIDADE DE SERVIÇO PARA REDES EM MALHA SEM FIO

Niterói
2009

BRUNO LIMA WANDERLEY

TC MESH: UMA FERRAMENTA DE GERÊNCIA DE QUALIDADE DE SERVIÇO PARA REDES EM MALHA SEM FIO

Dissertação apresentada ao Curso de Pós-Graduação "Stricto Sensu" em Engenharia de Telecomunicações da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre. Área de concentração: Sistemas de Telecomunicações.

Orientador: Profa. Dra. DÉBORA CHRISTINA MUCHALUAT SAADE

Niterói
2009

Ficha Catalográfica elaborada pela Biblioteca da Escola de Engenharia e Instituto de Computação da UFF

W245 Wanderley, Bruno Lima.
TC MESH : uma ferramenta de gerência de qualidade de serviço para redes em malha sem fio / Bruno Lima Wanderley. – Niterói,

RJ : [s.n.], 2009.

128 f.

Orientador: Débora Christina Muchaluat Saade.
Dissertação (Mestrado em Engenharia de Telecomunicações) - Universidade Federal Fluminense, 2009.

1. Rede Mesh. 2. Qualidade de serviço. 3. Sistemas de telecomunicação. I. Título.

CDD 004.3

BRUNO LIMA WANDERLEY

TC MESH: UMA FERRAMENTA DE GERÊNCIA DE QUALIDADE DE SERVIÇO PARA REDES EM MALHA SEM FIO

Dissertação apresentada ao Curso de Pós-Graduação “Stricto Sensu” em Engenharia de Telecomunicações da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre. Área de concentração: Sistemas de Telecomunicações.

Aprovada por:

Prof. Dr. Débora Christina Muchaluat Saade, TET/UFF

Prof. Dr. Célio Vinicius Neves de Albuquerque, IC/UFF

Prof. Dr. Alexandre Sztajnberg, IME/UERJ

Niterói
2009

A Deus e a todos aqueles que acreditaram que este dia chegaria.

Agradecimentos

Mais uma etapa de minha vida se encerra. É inevitável que para que eu chegasse ao fim deste ciclo contei com a ajuda de muitas pessoas. Primeiramente, gostaria de agradecer a minha família que ofereceu condições adequadas, tanto psicológicas quanto financeiras para finalizar o mestrado. Em segundo lugar, gostaria de agradecer a minha segunda família: a família MídiaCom. No qual sempre contei com a ajuda dos meus companheiros quando necessitei. Muito obrigado professora “mãe” Débora Saade, professor “chefe” Célio Albuquerque, professor “conselheiro” Luiz Magalhães pela oportunidade que me foi concedida de mostrar o meu valor, espero não ter decepcionado ninguém. Estou ciente que não sou nenhum gênio, mas usei o que eu considero que tenho de melhor para realizar minhas tarefas: disciplina e resiliência.

Aos companheiros Arthur Gerrante, Rafael Valle, Diego Passos, Joacir Silva, Jairo Duarte, Ricardo Carrano, Luciana Rocha e demais companheiros do projeto Remote. Muito obrigado pela companhia e por todo apoio que me foi oferecido.

Ao colega Álvaro Justen, por estar sempre solícito em ajudar na etapa final da dissertação. Sua ajuda foi muito importante.

Aos amigos Joacir Silva, William Mercado e Rafael Azevedo. Obrigado pelo apoio e amizade.

Aos colegas do projeto RUCA, pelo companheirismo.

A minha amável amiga e namorada, Eliana Brito (Ninha). Pelo apoio incondicional em todos os momentos desta etapa da minha vida.

Aos colegas Eunice Aguiar, Marcus, Daniel, Leandro, Francisco, Felipe, Eloy por compartilharmos juntos momentos de superação.

Enfim, muito obrigado a todos aqueles que direta ou indiretamente contribuíram para que este dia chegasse. Estou eternamente agradecido.

SUMÁRIO

p.

ÍNDICE DE FIGURAS

GLOSSÁRIO

ALTQ - *Alternate Queueing.*

ACK – *Acknowledgement.*

API - *Application Programming Interface.*

AODV – *Ad Hoc On-Demand Distance Vector.*

AWK - *Alfred V. **A**ho, Peter J. **W**einberger e Brian W. **K**ernighan.*

BGP – *Border Gateway Protocol.*

BSS – *Basic Service Set.*

CAC – *Connection Admission Control.*

CBQ – *Class Based Queueing.*

C-DATA – *Copy Data.*

CSMA/CD - *Carrier Sense Multiple Access / Collision Detect.*

CSMA/CA - *Carrier Sense Multiple Access /Collision Avoided.*

CTS - *Clear to Send.*

CFP – *Contention Free Period.*

DIFS – *DCF Interframe Space.*

DRSF – Dispositivo de rede sem fio.

DSNP - *Dynamic Service Negotiation Protocol.*

DSR – *Dynamic Source Routing Protocol.*

EDCA - *Enhanced Distributed Channel Access.*

EIFS - *Extended Inter-Frame Space.*

FIFO – *First in First Out.*

FTP – *File Transfer Protocol.*

G-DATA – *Get Data.*

GNU - *Gnu's Not Unix.*

HCF - *Hybrid Coodination Function.*

HFSC - *Hierarchical Fair Service Curve.*

HTB – *Hierarquical Token Bucket.*

IANA - *Internet Assigned Numbers Authority.*

IEEE - *Institute of Electrical & Electronics Engineers, Inc.*

IGMP - *Internet Group Management Protocol.*

JOBS - *Joint Buffer Management and Scheduling for Differentiated Services.*

LQSR - *Link Quality Source Routing.*

LDAP - *Lightweight Directory Access Protocol.*

MAC - *Media Access Control.*

MCL – *Mesh Connectivity Layer.*

MIB - *Management Information Base.*

MPU - *Minimal Packet Unit.*

MSN - *Microsoft Network.*

MTU - *Maximum Transmission Unit.*

MTV – *Mesh Topology View.*

NAT – *Network Address Translator.*

NAV – *Network Allocation Vector.*

NDIS - *Network Driver Interface Specification.*

OLSR – *Optimized Link State Routing.*

PBNM - *Policy Based Network Management.*

PC – *Personal Computer.*

PDP - *Policy Decision Point.*

PEP - *Policy Enforcement Point.*

PF – *Packet Filter.*

PIFS - *Priority Interframe Spacing.*

PHP - *PHP: Hypertext Preprocessor.*

PRIO - *MultiBand Priority Queueing.*

QAME - *QoS-Aware Management Environment.*

QGS - *QoS Global Server.*

QLN - *QoS Local Nodes.*

QoS – *Quality Of Service.*

RED – *Randon Early Dettection.*

RNP – Rede Nacional de Ensino e Pesquisa.

RREQ – *Route Request.*

RREP – *Route Reply.*

RRP - *Route and Reservation Reply.*

RRQ - *Route and Reservation Request.*

RSVP - *Resource Reservation Protocol.*

RTP – *Real Time Protocol*

RTS - *Request To Send.*

SFQ – *Stocastic Fair Queueing.*

SIFS - *Short Inter-Frame Space.*

SIP – *Session Initiation Protocol.*

SMNP – *Simple Network Management Protocol*.

SVG – *Scalable Vector Graphics.*

TBF – *Token Bucket Filter.*

TCNG – *Traffic Control Next Generation.*

TC MESH – *Traffic Control for Mesh Networks.*

U32 – *Universal 32 bits Comparisons.*

VoIP – *Voice over IP*.

XML - *Extensible Markup Language*.

RESUMO

Esta dissertação propõe uma ferramenta de gerência de serviço para redes em malha sem fio denominada TC MESH – Traffic Control for Mesh Networks. O TC MESH é uma ferramenta web que torna o processo de configuração e implantação de políticas de QoS em redes em malha sem fio mais rápido e fácil, pois oferece uma interface gráfica intuitiva para a definição dos parâmetros de QoS. Além disso, a ferramenta permite o monitoramento das políticas de QoS através da coleta de estatísticas da rede. Esta ferramenta foi testada e está em uso na rede em malha sem fio da Universidade Federal Fluminense na cidade de Niterói, RJ.

Palavras-chaves. Qualidade de Serviço, Redes Mesh, Ferramenta para Gerência de QoS.

ABSTRACT

This dissertation describes a quality of service management tool called TC MESH – Traffic Control for Mesh Networks. TC MESH is web-based and makes the configuration and deployment process of QoS policies in mesh networks much faster and easier, because it provides a simple graphical interface for the definition of QoS parameters. Moreover the tool allows the monitoring of QoS policies through the generation of statistics of the mesh network. This tool was tested and is currently in use in the wireless mesh network developed by the Fluminense Federal University in the city of Niterói, RJ.

Keywords. Quality of Service, Mesh Networks, QoS Management Tool.

1. INTRODUÇÃO

As redes sem fio IEEE 802.11 de modo geral podem ser consideradas hoje como tendência mundial, graças à sua popularização. Diversas implantações têm sido feitas em universidades, escolas e até mesmo dentro dos lares da população. Especificamente, as redes em malha sem fio (redes *mesh*) surgiram a partir de redes *ad-hoc* e se caracterizam por possuírem seu núcleo (*backbone*) formado por uma rede sem fio em que os elementos se interconectam. A partir desse núcleo, os usuários podem ter acesso à Internet através de *gateways*. Uma rede em malha sem fio também permite a conexão de dispositivos móveis através de uma interface sem fio, permitindo o acesso a Internet em banda larga. Porém, o crescimento do número de nós que compartilham o enlace sem fio é diretamente proporcional ao decréscimo do desempenho geral devido ao aumento da competição pelo meio físico. característica inerente da tecnologia. Desta forma, a quantidade de saltos (*hops*) necessária para que uma transmissão chegue até o nó de destino afeta significativamente o desempenho da transmissão. Assim sendo, cria-se a necessidade de obter alternativas para oferecer melhor desempenho para determinadas aplicações e diferenciação dos serviços e assim atenuar os efeitos inerentes da tecnologia.

Em redes sem fio, os enlaces de rádio estão sujeitos a interferências, colisões e atenuações do sinal devido a obstáculos. As redes *ad hoc* e *mesh*, além de serem redes sem fio, permitem a mobilidade de alguns de seus nós, o que gera mudança da topologia da rede e consequentemente, ocasiona problemas de conectividade entre seus enlaces. Além disso, muitos nós que compõem a rede apresentam limitações nos recursos disponíveis, incluindo alimentação elétrica e capacidade de transmissão.

Garantir qualidade de serviço em redes em malha em fio não é uma tarefa simples. Os mecanismos desenvolvidos devem ter como objetivo agilizar o processo de adaptação da rede às necessidades do usuário naquele momento de forma rápida e prática para um uso mais eficiente e completo dos recursos. Para isso, os serviços disponíveis na rede, além de serem capazes de distinguir entre diferentes requisitos de qualidade especificados pela aplicação, devem ajudar a superar os problemas causados pelas características do meio sem fio, como instabilidade da qualidade dos enlaces. Oferecer qualidade de serviço é uma tarefa essencial em redes *mesh* devido às diferentes demandas das aplicações .

A rede em malha sem fio do projeto ReMesh [REMESH, 2009], desenvolvido pela Universidade Federal Fluminense (UFF) teve como objetivo implantar uma rede sem fio de acesso comunitário em um dos campi da UFF com o intuito de fornecer, a baixo custo, acesso banda larga para a comunidade acadêmica que reside ao redor do campus. No projeto ReMesh são usados roteadores que possuem o sistema operacional *OpenWRT* [OPENWRT, 2009], tratando-se de uma versão reduzida do sistema operacional GNU/Linux. Através desse sistema, pode ser usada a ferramenta de controle de tráfego do GNU/Linux (*Traffic Control* - TC) [HUBERT, 2009] para a configuração de parâmetros de qualidade nos roteadores, sendo possível o uso de diversas disciplinas de enfileiramento, criação de classes com suas respectivas prioridades e filtros (classificadores) para cada classe. Cada disciplina de enfileiramento tem um comportamento único que deve ser analisado de acordo com as características do tráfego que atravessará o nó da rede. Existem disciplinas que se adaptam melhor para tráfegos multimídia, reduzindo o atraso total da comunicação e outras adequadas a vários tipos de tráfego (FTP, por exemplo). O conceito de classes oferece a possibilidade de ramificação do tráfego de dados entrante para diversas filas de pacotes em cada roteador de acordo com a prioridade estabelecida, permitindo assim diferenciar os fluxos e oferecer características diferentes para cada um, seja uma vazão maior, ou maior prioridade, disciplinas de enfileiramento específicas, entre outras. Embora o TC ofereça uma grande variedade de funções, o seu manuseio não é trivial, necessitando de um período de aprendizagem do uso das principais funções do *software*, pois, é necessário o desenvolvimento de scripts com a sintaxe do TC para a aplicação dos parâmetros. Além disso, o tempo gasto no desenvolvimento do script com os parâmetros de QoS e a sua aplicação em todos os roteadores da rede torna-se uma tarefa muitas vezes demorada, reduzido o tempo para análise dos resultados devido ao excessivo tempo gasto no desenvolvimento, cópia, execução e análise das políticas. A necessidade de uma adaptação rápida da rede de acordo com o fluxo de dados faz com que a agilidade na mudança das políticas de QoS seja algo crítico para o oferecimento de um serviço com a melhor qualidade possível.

Tendo em vista os fatores citados com relação ao desenvolvimento e execução das políticas, este trabalho tem como objetivo o desenvolvimento da ferramenta "*Traffic Control for Mesh Networks*" (*TC MESH*). A ferramenta oferece ao usuário a possibilidade de criação do script, que é um arquivo texto com a sintaxe do TC, com as políticas de QoS desejadas sem a necessidade de qualquer espécie de programação e comunicação direta com os roteadores. O *TC MESH* oferece uma interface gráfica web onde o usuário define os parâmetros necessários de maneira intuitiva. O único pré-requisito para o uso do *TC MESH* é

o conhecimento básico dos fundamentos de redes de computadores e qualidade de serviço. Apesar disso, o usuário fará melhor uso da ferramenta se obtiver informações mesmo que básicas do funcionamento do controle de tráfego do GNU/Linux, oferecido no guia do usuário no site da ferramenta. Este guia é de fácil entendimento e rápida leitura para os profissionais de tecnologia da informação. Além da possibilidade da criação das políticas, o *TC MESH* oferece a automatização do processo de instalação e execução dos scripts nos roteadores, bastando ao usuário definir através de uma barra de opções para quais roteadores se deseja que aquela política seja aplicada. Uma outra funcionalidade da ferramenta é a possibilidade da visualização gráfica das estatísticas do tráfego de cada classe em cada roteador. Com as funcionalidades do *TC MESH*, o tempo gasto pelo usuário para desenvolver o script, colocá-lo em operação e analisar os resultados é reduzido drasticamente.

O principal objetivo da ferramenta *TC MESH* é facilitar e incentivar o uso das políticas de qualidade de serviço nas redes em malha sem fio, como por exemplo a rede mesh da UFF implantada pelo laboratório MídiaCom. A partir do *TC MESH*, os procedimentos de implantação (desde a criação das políticas até a coleta de estatísticas da rede) e análise serão significativamente mais rápidos, permitindo aos usuários utilizarem o tempo excedente para uma maior análise dos dados coletados e consequentemente obter conclusões mais consistentes sobre os testes de QoS realizados. Além disso, o aprendizado prático em QoS ajuda na formação da equipe administradora da rede permitindo a contextualização da teoria com os testes práticos.

1.1. ESTRUTURA DA DISSERTAÇÃO

Esta dissertação está organizada como segue. O Capítulo 2 comenta conceitos importantes em redes sem fio IEEE 802.11, com ênfase em redes em malha, citando alguns exemplos de propostas de gerência de QoS. Ainda neste capítulo são descritos os principais desafios para a provisão de qualidade de serviço em redes sem fio de modo geral. O Capítulo 3 descreve alguns aplicativos de gerência de QoS citados na literatura. Já no Capítulo 4, são exemplificadas as principais características com relação a disciplinas de enfileiramento, classes e filtros do controle de tráfego do GNU/*Linux*. A ferramenta *TC MESH* é então descrita no Capítulo 5, detalhando o seu funcionamento e as vantagens da mesma com relação a outras ferramentas. O Capítulo 6 apresenta as conclusões da dissertação juntamente com as contribuições e trabalhos futuros.

2. QUALIDADE DE SERVIÇO EM REDES EM MALHA SEM FIO

Neste capítulo, serão descritas as principais premissas em relação a redes sem fio IEEE 802.11 com ênfase nos parâmetros relevantes à qualidade de serviço. Além disso, serão descritos alguns projetos de gerência/administração de QoS usados no meio acadêmico e os principais desafios no provimento de qualidade em redes em malha sem fio. Apesar da ferramenta de gerência de QoS *TCMesh* utilizar predominantemente conceitos da camada de rede, é importante ressaltar alguns aspectos pertinentes à infraestrutura de enlace utilizada em uma rede em malha sem fio.

2.1. O PADRÃO IEEE 802.11

O padrão para redes sem fio IEEE 802.11 foi criado em 1997 com seus primeiros suplementos incluídos em 1999, tratando dos padrões 802.11a e 802.11b. Em 2003, o 802.11g foi formalmente ratificado pela IEEE Standarts Association´s Board [IEEE, 2009]. Similar ao padrão Ethernet (IEEE 802.3), IEEE 802.11 oferece uma camada física otimizada e uma camada de controle de acesso ao meio (MAC) como uma subcamada para o oferecimento de serviços de comunicação de dados em redes locais sem fio.

Funcionando diferentemente do controle de acesso ao meio CSMA/CD (Carrier Sense Multiple Access/Collision Detect) [KUROSE e ROSS, 2004] usado no Ethernet, o 802.11 usa o CSMA/CA (Carrier Sense Multiple Access/Collision Avoided) [KUROSE e ROSS, 2004] devido às características de um ambiente de rede sem fio. Os tradicionais problemas de terminal escondido, onde dispositivos sem fio não percebem a colisão de dados enquanto estão transmitindo (vide Figura 1), e terminal exposto, que é aquele que está ao alcance de um transmissor mas fora do alcance do receptor (vide Figura 2) , inviabilizam o uso do protocolo CSMA/CD em redes sem fio. No padrão IEEE 802.11, os quadros devem ser retransmitidos se a estação de origem não receber a confirmação do recebimento dos dados (ACK) enviada pelo destino.

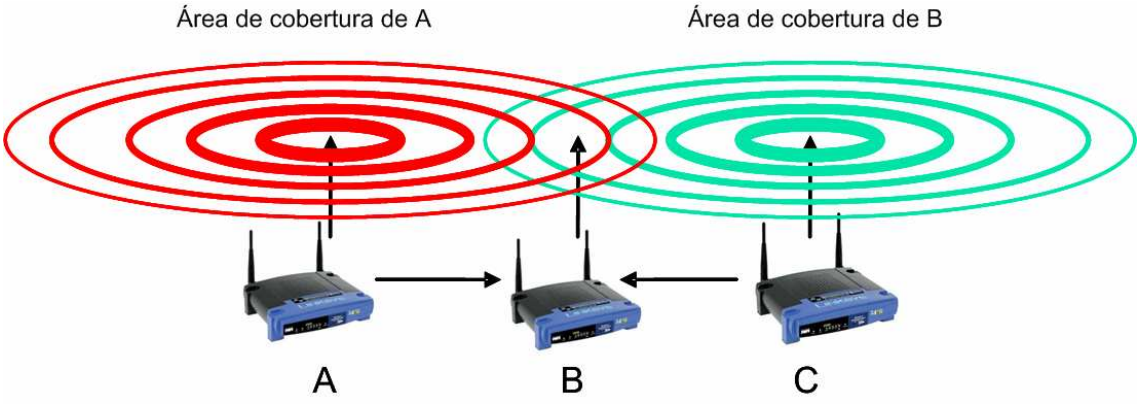

Figura 1. Exemplo de terminal escondido em uma rede sem fio.

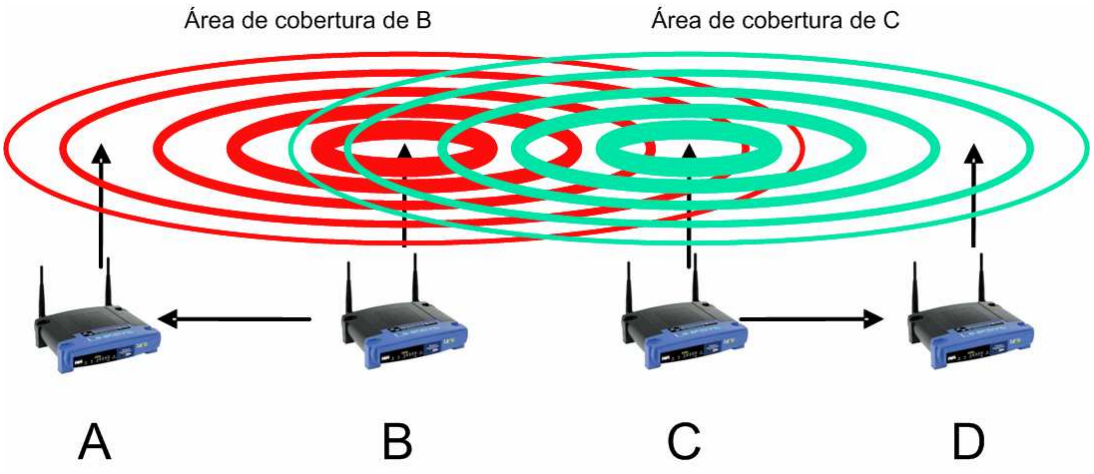

Figura 2. Exemplo de terminal exposto em uma rede sem fio

O padrão IEEE 802.11 define que há dois protocolos possíveis para acesso ao meio físico de um conjunto de estações que se comunicam em um ambiente sem fio, denominado *Basic Service Set* (BSS): *Distributed Coordination Function* (DCF) e *Point Coordination Function* (PCF). A extensão ao padrão IEEE 802.11e especifica outro mecanismo de acesso chamado HCF (*Hybrid Coordination Function*), que oferece suporte a transmissão de diferentes tipos de tráfego.

2.1.1. DCF

O DCF (*Distributed Coordination Function*) é um esquema de acesso ao meio distribuído baseado no CSMA/CA. Neste modo, um dispositivo de rede sem fio deve saber o estado do meio físico antes de iniciar uma transmissão de pacotes. Dois modos de análise do meio são permitidos: analisando o meio a partir da interface física ou de forma virtual usufruindo da camada MAC. Analisando o meio através do nível físico é possível detectar a

presença de outras interfaces de rede analisando todos os pacotes e a verificação da atividade de cada canal do padrão em questão, detectando a partir da potência do sinal das interfaces de rede vizinhas [PEREIRA e WESTPHALL, 2005]. A análise do sinal realizada virtualmente pode ser usada para que um dispositivo de rede sem fio notifique aos outros dispositivos vizinhos dentro do mesmo BSS quanto tempo o canal será reservado para transmissão de quadros. Nesta proposta, o emissor dos dados pode informar um tempo no cabeçalho MAC, ou usar os quadros de controle *RequestToSend* (RTS) e *ClearToSend* (CTS) para a transmissão de quadros. Desta forma, outras interfaces de rede podem atualizar seus vetores de alocação de rede (NAV - *Network Allocation Vectors*) para indicar essa duração. Como mostrado na Figura 3, se o meio físico se encontra sem tráfego em um intervalo de tempo maior que um *Distributed Interframe Space* (DIFS), a interface de rede de origem transmitirá imediatamente o quadro. Enquanto isso, outras estações aguardam o momento das suas transmissões e ajustam seus NAV's, e o processo de *backoff* começa. Neste processo, o dispositivo de rede computa um intervalo de tempo aleatório, chamado *backoff timer*, selecionado a partir da janela de contenção (*ContentionWindow* - CW):

$$backoff\_timer = random\ [0, CW] * slot\_time \quad (1)$$

Onde CWmin<CW<CWmax e *slot time* dependem do tipo de camada física em uso. O *backoff timer* é reduzido somente quando o meio físico está livre sendo congelado quando outra interface de rede transmitir. A cada momento que o meio fica livre, o dispositivo de rede sem fio (DRSF) espera por um DIFS e decrementa continuamente o tempo de *backoff*. Quando o tempo de backoff expira, o DRSF é autorizado a ter acesso ao meio. Logicamente, colisões irão ocorrer se dois ou mais DRSF's acessarem o meio ao mesmo tempo. Diferentemente de uma rede cabeada, a detecção de colisão durante a transmisssão em uma rede sem fio torna-se impossível devido a uma significativa diferença entre a potência do sinal enviado e recebido. Então, uma resposta positiva de ACK é usada para notificar o emissor que o quadro enviado foi recebido com sucesso, como visto na Figura 3(A). O não recebimento do ACK pelo emissor indica que o quadro sofreu colisão e a retransmissão é realizada com o processo de *backoff* reiniciado. Para reduzir a probabilidade de colisões, após cada transmissão mal sucedida o CW é dobrado até o valor CWmax ser alcançado. Após cada transmissão bem sucedida o CW vai ser diminuído até seu valor CWmin.

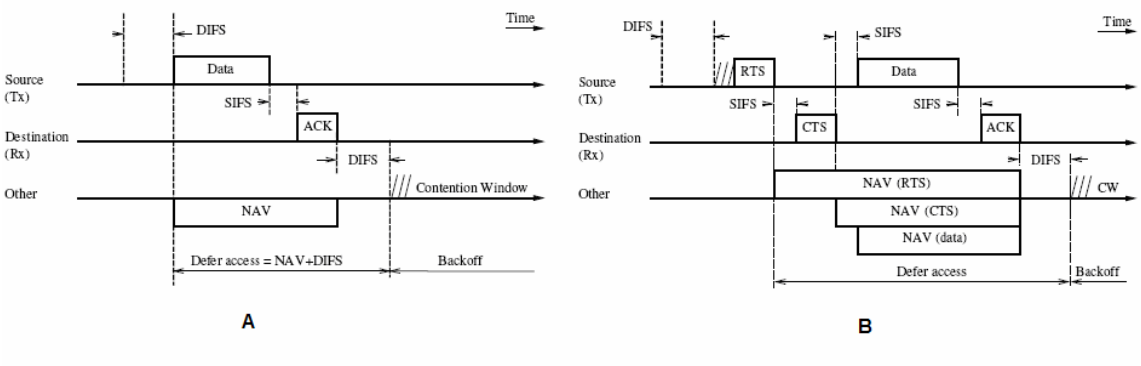

Figura 3. Envio de quadros sem e com RTS/CTS [PEREIRA e WESTPHALL, 2005].

Terminais escondidos são DRSF´s que podem 'escutar' o meio mas não podem ser detectados por outros dispositivos. Consequentemente, os pacotes de diferentes emissores colidirão em um mesmo receptor. Para resolver este problema, usa-se o esquema chamado RTS/CTS. Tal esquema faz com que a origem envie um quadro RTS (*Request To Send*) antes de cada transmissão, como mostra a Figura 3(B), e o receptor responde com um CTS (*Clear To Send*), se estiver pronto para receber. Depois que a origem receber o CTS, é iniciada a transmissão do quadro de dados. Então, todas as DRSF's que escutaram um RTS, ou um CTS ou até mesmo um quadro podem atualizar suas NAV's e não iniciar transmissões naquele intervalo. O esquema RTS/CTS melhora o desempenho de um esquema DCF consideravelmente em alguns casos [RUBINSTEIN e REZENDE, 2002]. Deve-se levar em consideração o *overhead* que esse esquema gera caso o tamanho dos quadros enviados seja pequeno, aumentando significativamente o tráfego na rede. Existem trabalhos [HIDEKI et al, 2007] que comprovam a ineficiência do RTS/CTS em topologias específicas como a linear, onde os roteadores se comunicam entre si com antenas direcionais e com o nível de interferência entre os roteadores reduzido.

2.2. PCF

O PCF (*Point Coordination Function*) é o outro tipo de acesso da subcamada MAC do 802.11, de implementação opcional. No PCF, um coordenador da rede faz o controle do acesso ao meio através de consulta a cada estação, possibilitando a oportunidade de transmitir sem contenção. O coordenador da rede (ponto de acesso em uma rede sem fio infraestruturada) divide o tempo de acesso em períodos de superquadros. O superquadro compreende num período livre de contenção (modo PCF) e um período com contenção (DCF). Quando as estações estão em um período PCF, o coordenador verifica se cada estação precisa

transmitir algo. Cada estação recebe dados quando o coordenador consulta-as. O coordenador escuta o meio por um intervalo PIFS (*Point Coordination Interframe Space*) e logo depois começa um período livre de contenção (*Contention Free Period – CFP*) através da difusão de um sinal de *beacon* [PEREIRA e WESTPHALL, 2005].

Conceitualmente, PIFS é menor que DIFS, então nenhuma estação pode começar a enviar dados com o modo DCF durante o período sem conteção. As estações adicionam a duração máxima do período sem contenção aos seus respectivos NAVs. Esse período livre de contenção pode terminar com o envio de uma sinalização (CFend) pelo coordenador. Quando chega a vez de uma estação transmitir, o coordenador da rede envia um quadro de dados se existir algum a ser enviado dentro de um quadro de consulta [PEREIRA e WESTPHALL, 2005]. É enviado de volta um ACK pelo receptor também com dados se for o caso, depois de SIFS segundos. Encerrando a transmissão a todas as estações contidas em uma lista de consultas, o coordenador reinicia o processo de consulta depois do intervalo de tempo PIFS. Se houver usuários que estão sem transmitir por alguns ciclos, eles são removidos de lista de consultas e são consultados de novo no início do próximo período livre de contenção.

No modo PCF, os mecanismos de QoS têm a ver com o tipo de escalonamento associado à interrogação das estações. Em vez das estações serem interrogadas num mecanismo *Round-Robin* ou qualquer outro puramente justo, o AP pode estabelecer um mecanismo *Weighted Fair Queueing* (WFQ), em que pode-se atribuir um peso a cada estação de acordo com a prioridade devida aos requisitos de QoS solicitados [RUBINSTEIN e REZENDE, 2002].

Qualquer que seja o mecanismo a considerar para suportar QoS em PCF temos sempre que ter em conta os obstáculos apresentados anteriormente em relação ao PCF: não ser correntemente suportado pela maioria das implementações, não ser escalável, ser limitado no caso de múltiplas BSSs.

2.2.1. IEEE 802.11E

O padrão IEEE 802.11e [RUBINSTEIN e REZENDE, 2002] especifica um novo método de coordenação chamado de HCF (*Hybrid Coordination Function*), que provê os modos de acesso com e sem contenção. No acesso ao meio baseado em contenção, chamado de EDCF (*Enhanced* DCF), cada fluxo usa de diferentes DIFS e CWmin. Já com o acesso sem contenção HCCA (*Hybrid Coordination Function Controlled Channel Access*), técnicas foram incluídas para permitir a diferenciação de tráfego. Como tal método exige a presença

de um ponto de acesso, este mesmo é o responsável por prover essa diferenciação. As técnicas podem ser classificadas de acordo com o mecanismo empregado, que é baseado na consulta e na alocação de *slots* de tempo. As técnicas baseadas em consulta às estações levam em conta a prioridade destas estações. Em técnicas baseadas em alocação de slots, o ponto de acesso determina, também por meio de consulta, os slots de tempo atribuídos a cada uma das estações.

Depois que esta atribuição é feita, esta técnica não precisa mais de nenhum tipo de intervenção do ponto de acesso. O IEEE 802.11e define um acesso baseado em consultas, onde o ponto de acesso estabelece um intervalo de tempo no qual uma determinada estação pode iniciar a transmissão. Tais oportunidades de transmissão podem ser criadas nos períodos com ou sem contenção. Como o padrão IEEE 802.11e é mais recente, sua utilização prática em redes sem fio não é muito comum.

2.3. REDES EM MALHA SEM FIO

As redes em malha sem fio são redes com topologia dinâmica, variável e de crescimento orgânico, constituídas por nós cuja comunicação, no nível físico, é feita através de uma das variantes do padrão IEEE 802.11, e cujo roteamento é dinâmico. Essas redes têm muito em comum com as redes móveis *ad-hoc* (Mobile Ad-hoc networks, ou MANETs) [ABELÉM et al, 2007]. Assim com estas últimas, as redes *mesh* utilizam meio de transmissão sem fio. O que diferencia principalmente as duas é o fato de que os nós das redes em malha sem fio têm localização fixa, mesmo que suas localizações não sejam predeterminadas. Os algoritmos de roteamento, portanto, têm muita semelhança entre si. Na Figura 4, vemos um exemplo de como poderia ser empregada uma rede em malha sem fio. Os nós sem fio comunicam-se entre si, roteando o tráfego, que é escoado através de um *gateway* para a Internet.

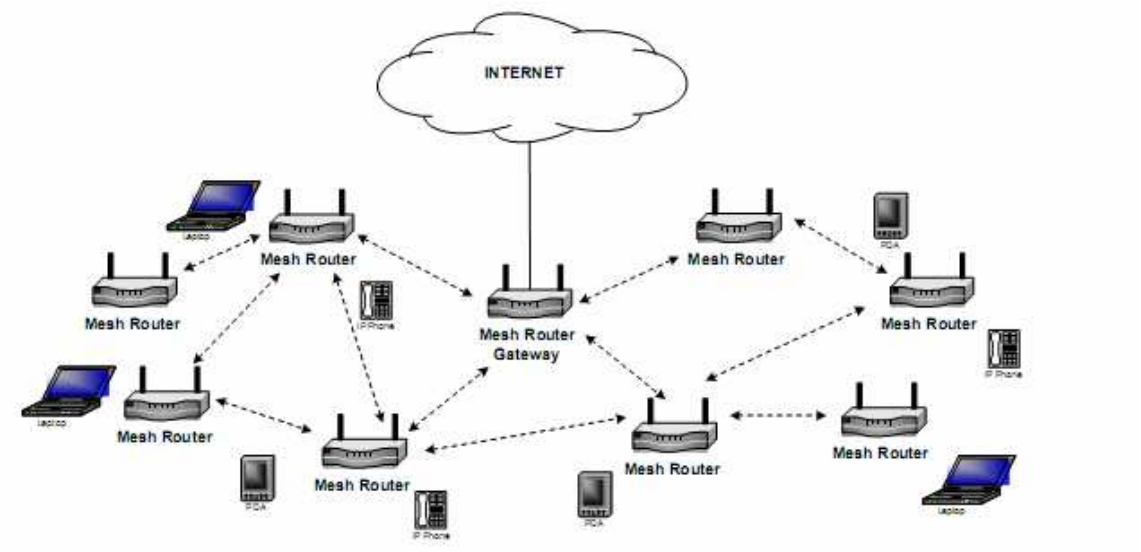

Figura 4. Exemplo de rede em malha sem fio [ABELÉM et al, 2007].

Entre as principais vantagens das redes em malha sem fio está a já citada racionalização de custos. O custo dos enlaces sem fio é mais baixo que o de criar novos pontos de acesso com fio, que podem incluir a necessidade de obras caras, trabalhosas e inconvenientes. E o custo do hardware cai rapidamente. Já existem à venda no mercado equipamentos apelidados "*mesh boxes*". Estes equipamentos encapsulam, em caixas pequenas e resistentes, todo o hardware e software necessário para criar um nó de uma rede em malha sem fio. Estes kits incluem tipicamente um computador, uma ou mais antenas, um sistema operacional (na maioria dos casos, um sistema aberto, como Linux) e o *software* de roteamento específico.

A simplicidade para o usuário final é outra vantagem deste tipo de rede. Como o roteamento é dinâmico (e, normalmente, a atribuição de endereços também), a instalação resume-se a ligar o *mesh box* à tomada, ou instalar um software no PC do usuário. O roteamento dinâmico também significa que dificilmente será necessário um administrador da rede. As condições da rede mudam como a adição ou remoção de um nó ou interferência em um enlace e a rede se adapta sem interferência humana. Esta capacidade de adaptação automática é outra vantagem das redes em malha sem fio. O termo "malha" (ou *mesh*) implica que a maioria dos nós estará conectada a mais de um outro nó. Com isso, a rede torna-se muito mais robusta que uma rede convencional, pois a queda de um nó ou de um enlace não implica em quebra da conectividade, já que o sistema é capaz de encontrar novas rotas dinamicamente [ABELÉM et al, 2007].

O projeto ReMesh [REMESH, 2009] da Universidade Federal Fluminense desenvolveu um protótipo de nó *mesh* de baixo custo encapsulando dentro de uma caixa hermética o roteador Linksys modelo WRT54GL [WRT54GL, 2009] com configuração *mesh* em uso nos arredores do campus da Escola de Engenharia da UFF, em Niterói. O projeto ReMesh utiliza o protocolo de roteamento OLSR (*Optimized Link State Routing*) com a métrica ML (*minimum loss*) [MUCHALUAT et al, 2007].

O Laboratório MídiaCom da UFF está também desenvolvendo através do projeto REMOTE (Rede de Monitoramento para linhas de Transmissão de Energia) [REMOTE, 2009] uma rede em malha sem fio linear, situada no topo de torres de linhas de transmissão de energia elétrica no estado de Santa Catarina com extensão de aproximadamente 50 Km. Este projeto tem como objetivo principal demonstrar a viabilidade de uma rede de comunicação banda larga do tipo *mesh* para supervisão e comunicação ao longo de linhas de transmissão de energia situadas em locais de limitada infraestrutura de comunicação. Com esta rede será oferecida conectividade para equipes de manuntenção de campo, viabilizando a utilização de sensores diversos além de fluxos de dados multimídia com qualidade de serviço. No projeto REMOTE, os roteadores sem fio são alimentados através de energia solar, pois são instalados em torres de alta tensão.

2.4. PROPOSTAS DE QOS EM REDES EM MALHA SEM FIO

Vários esforços para oferecer qualidade de serviço em redes sem fio têm sido propostos ao redor do mundo. Cada projeto tem caminhos diferentes para a provisão de QoS em redes sem fio, alguns visam uma abordagem no nível de enlace, já outros no nível de rede e transporte.

Em [HAMIDIAN e KOLTER, 2007], é descrito um mecanismo de contenção da camada de acesso ao meio do padrão IEEE 802.11e, o EDCA, que foi aprimorado para permitir que estações possam reservar o meio para aplicações multimídia com necessidades de QoS. O esquema proposto trata-se do EDCA com reserva de recursos (EDCA/RR), descrevendo de que forma o mesmo pode ser implantado. O EDCA/RR opera de maneira distribuída e gerencia a rede para prover maior liberdade para que serviços multimídia usufruam do meio físico.

EDCA/RR foi aprimorado para ser usado em uma rede que permita múltiplos saltos. Ao invés da rede iniciar com o processo de descobrimento de rotas e depois com a etapa de reserva de recursos, faz-se uma combinação dos dois procedimentos. A vantagem desta

abordagem é que no descobrimento de rotas a rede já estará ciente dos requisitos de QoS da aplicação e, consequentemente, a estação irá procurar a rota que atende as necessidades da aplicação. Outra vantagem é menor atraso e *overhead* na descoberta e alocação de recursos.

Quando o primeiro pacote de alta prioridade chega ao protocolo de roteamento *ad hoc* ele checa na tabela uma rota para o destino. Se o pacote for identificado como um fluxo multimídia, o protocolo de roteamento AODV (*Ad-Hoc On-Demand Distance Vector*) [IAN et al, 2004] iniciará o processo de descoberta/reserva de recursos para gerar uma requisição de rota e reserva (RRQ – *Route and Reservation reQuest*). A mensagem RRQ é uma requisição de rota (RREQ – *Route REQuest*) adicionada de alguns campos como a taxa de transmissão e a prioridade do pacote. Estes parâmetros devem ser ajustados pela aplicação. A mensagem RRQ é enviada para a camada de enlace de dados e assim a subcamada MAC inicia o procedimento de reserva de recursos checando se o novo fluxo tem o perfil adequado para ser admitido. O EDCA/RR indica apenas se o tráfego deve ser admitido ou rejeitado, mas não reserva recursos neste etapa. A razão é que até este ponto a estação não sabe se a rota (multissaltos) entre a origem e o destino realmente existe ou não – e mesmo se tal rota existir, a estação não sabe se há recursos disponíveis na rede para realizar a reserva.

Cada estação tem conhecimento sobre reservas a até dois saltos entre seus vizinhos. Consequentemente, a reserva de recursos é realizada quando um determinado destino envia uma resposta para a origem (depois que o RRQ indicar que há recursos suficientes na rede) com informações de todos os nós por que passará o pacote. Se o controle de admissão falhar, a aplicação pode preferir excepcionalmente uma rota sem QoS já que não há nenhuma rota que tenha os requisitos mínimos. Embora algumas aplicações necessitem de certos requisitos de QoS para funcionar adequadamente, outras podem funcionar mesmo que não haja o nível de QoS adequado. Esta opção fica a cargo da aplicação, indicando se aquele fluxo aceita ou não a transmissão sem QoS. Por outro lado, se o controle de admissão é bem sucedido, o RRQ é enviado via broadcast como de costume. A estação que receber a mensagem guarda a informação sobre a reserva e inicia o procedimento de reserva de recursos na subcamada MAC para checar se o novo fluxo pode ser admitido. Se o fluxo puder ser admitido então a mensagem é encaminhada para todas as estações intermediárias até ser recebida pelo destino. O destino responde com uma resposta de rota e reserva (RRP – *Route and Reservation Reply*), que é semelhante a uma tradicional resposta do roteador (RREP – *Route Reply*) adicionada de informações sobre reserva de recursos. Esta mensagem confirma a requisição de reserva e muda o status da rota de “admissível” para “reservada”. As estações escutarão essa mensagem

e ficarão cientes dos recursos da rede. Desta forma, é oferecido um melhor serviço para os fluxos mais sensíveis às variações da rede.

O QMesh [BORTINKOV et al, 2007] é um *software* que permite a utilização de múltiplas áreas de trabalho com o Sistema Operacional *Windows* com uma infraestrutura de redes em malha sem fio com suporte a mobilidade. O QMesh dá suporte a seus usuários através dos protocolos tradicionais de rede sem fio e não necessita de instalação de nenhum software cliente. A solução otimiza a provisão de QoS, oferecendo um gerenciamento centralizado da infraestrutura, permitindo uma transferência de recursos entre os usuários de acordo com a prioridade e a distância até o *gateway* do *desktop*.

O QMesh é implementado no *kernel* do sistema operacional *Windows* XP, no topo da Camada de Conectividade Mesh (*Mesh Connectivity Layer* – MCL) desenvolvida pela *Microsoft Research* que provê roteamento básico dentro da rede. O MCL implementa o protocolo de roteamento LQSR, que trata-se de uma versão modificada do DSR [BORTINKOV et al, 2007] que implementa métricas sofisticadas na camada de enlace para roteamento de redes sem fio com múltiplos saltos. Adicionalmente o MCL oferece a funcionalidade de servir como uma infraestrutura de acesso a redes sem fio, que provê micromobilidade entre os *Access Points mesh*. Com relação à arquitetura da rede, o código MCL na verdade é um driver NDIS (*Network Driver Interface Specification*) que consiste basicamente de uma API (*Application Programming Interface*) para interfaces de rede. É o NDIS que implementa LQSR entre a camada de enlace de dados e rede. O QMesh adiciona a este driver a possibilidade de resolução do endereço MAC e o encaminhamento do tráfego unicast/multicast para usuários que não usufruem do LQSR. O QMesh controla o usuário que funciona como *Access Point* e *Gateway* através das extensões do LQSR. A Figura 5 ilustra a arquitetura do QMesh.

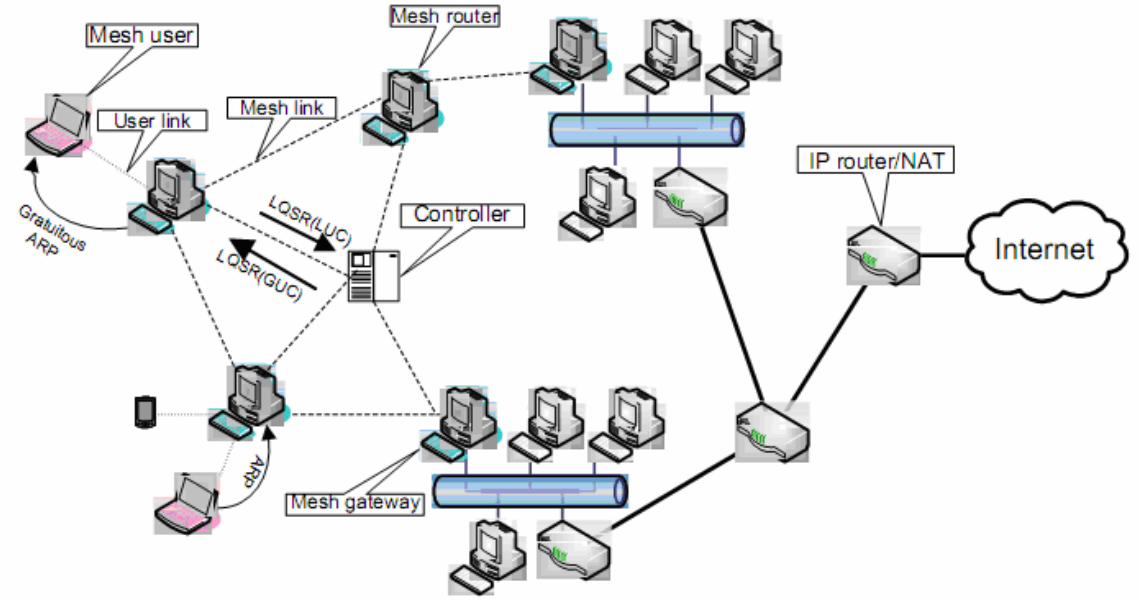

Figura 5. Arquitetura do QMesh [BORTINKOV et al, 2007].

O QMesh associa cada usuário com um único AP. Entretanto, não é possível implementar nos AP's pontes transparentes como no modo infraestruturado do 802.11, devido a uma peculiaridade da maioria dos drivers das placas de rede sem fio no ambiente Windows não oferecerem suporte para o uso em modo promíscuo. Usuários móveis então necessitam se comunicar diretamente com os AP's, através do modo ad hoc do 802.11.

Embora os dois trabalhos citados anteriormente auxiliem no provimento de QoS para redes de computadores, eles não oferecem ao usuário a facilidade de escolher que tráfego receberá diferenciação dos serviços. Em [HAMIDIAN e KOLTER, 2007], o procedimento de reserva dos recursos acarreta maior sobrecarga na rede. Além disso, os recursos necessários para serviços menos prioritários poderão não estar disponíveis por muito tempo. Uma abordagem alternativa mais eficiente seria o uso da diferenciação dos serviços auxiliada por um controle de admissão em cada classe com a finalidade de bloquear certos fluxos, caso não haja recursos disponíveis em determinado momento. Logicamente, o controle de admissão bloquearia mais fluxos menos prioritários.

2.5. DESAFIOS PARA A PROVISÃO DE QOS EM REDES EM MALHA SEM FIO

A qualidade de uma rede *mesh* depende fundamentalmente da qualidade dos enlaces sem fio, do nível de interferência e da taxa de utilização dos recursos de rádio. As redes *mesh* têm diversos aspectos que tornam sua análise de desempenho mais complexa que as redes sem fio tradicionais. Alguns desses aspectos são a competição entre o tráfego originado no nó *mesh* e o encaminhado pelo nó *mesh*, o decréscimo da banda efetiva com o aumento do número de saltos até o destino e a disputa pelo acesso ao meio de pacotes de um mesmo fluxo entre nós vizinhos. Entretanto, é possível oferecer diferenciação dos serviços a partir de *softwares* que agem na camada de rede, diferenciando certos tipos de fluxos mais sensíveis a atrasos e vazão, como voz e vídeo. A partir de uma análise dos aspectos que serão descritos a seguir é possível dimensionar o quanto esses fatores podem deteriorar a qualidade dos enlaces, oferecendo uma estimativa da capacidade média de fluxos na rede. A ferramenta *TC MESH* pretende solucionar alguns destes problemas. Mais especificamente aqueles referentes ao nível de rede, camada na qual a ferramenta atua. Nos tópicos a seguir serão descritos os principais problemas e de que forma a ferramenta pode solucionar ou atenuar alguns deles.

2.5.1. COMPETIÇÃO INTRA-ROTEADOR E INTER-ROTEADOR

Para garantir os requisitos de QoS de um pacote em uma rede sem fio, diferenciar serviços dentro do roteador não é totalmente eficiente, através do gerenciamento de filas e de algoritmos de escalonamento, como ocorre nos meios com fio. Para chegar ao seu destino [CHEN et al, 2000], o pacote precisa disputar a saída do nó através do esquema de prioridades de filas e escalonamento de serviço das filas apropriados, porém deve disputar também o acesso ao meio a cada salto, muitas vezes contra quadros do mesmo fluxo, impactando na QoS fim-a-fim da aplicação de forma geral. A ferramenta *TC MESH* através do controle de tráfego do GNU/Linux permite o gerenciamento de filas em cada roteador, contribuindo para o planejamento da competição intra-roteador. Além disso, através da visualização das estatísticas de vazão de cada fila, é possível analisar o comportamento da rede de acordo com as políticas de filas e realizar uma relação da capacidade do enlace *versus* a quantidade de saltos da rede, verificando a vazão em um deteminado enlace ou em vários saltos. Assim, a implantação das políticas de QoS em nível de rede pode ajudar a atenuar os problemas de desempenho da rede usando-a da maneira mais eficiente possível.

2.5.2. CAPACIDADE DA REDE

Um aspecto relevante em redes em malha sem fio é, sem dúvida, a capacidade de uma rede mesh devido a sua redução de banda ser diretamente proporcional ao número de saltos (*hops)*. A seguir, serão descritos os principais fatores relevantes sobre o problema:

- ✓ Números de nós na rede: É recomendável para um desempenho satisfatório que a soma das vazões por nó não ultrapassem a capacidade do *gateway*, pois poderá ocorrer da carga na rede crescer de acordo com o número de nós, aumentando a disputa pelo meio físico.
- ✓ Densidade de nós e sua variação ao longo da rede.
- ✓ Posição dos nós e carga oferecida pelos demais nós: Quando os nós que sofrem menos interferência inserem mais tráfego na rede do que seus vizinhos são capazes de repassar.
- ✓ Localização e quantidade de *gateways*: Parâmetro relacionado com a escalabilidade da rede.
- ✓ Em redes de múltiplos saltos, os problemas do “terminal escondido” e do “terminal exposto” são agravados, aumentando a interferência entre eles e piorando o desempenho da rede.

2.5.3. GERÊNCIA DE BANDA EFICIENTE E ADAPTATIVA

Existem momentos onde diferentes classes de tráfego requisitam reserva de largura de banda, sendo tarefa do algoritmo de controle de tráfego oferecer os recursos de acordo com as necessidades e prioridades de cada classe [GALLEGO e ALFONSO, 2005]. Esquemas otimizados com a função para a marcação dos fluxos de acordo com as suas necessidades de QoS, como ocorre no TC, fazem com que este tráfego atravesse a rede adequadamente. É importante ressaltar que a análise do tipo de fluxo é muito importante. No caso de um fluxo multimídia em tempo real devem ser levados em consideração não somente a vazão destinada aquele fluxo, mas também parâmetros de *jitter* e atraso fim-a-fim, a fim de permitir que o tráfego flua de forma que não degrade a qualidade do fluxo. A ferramenta *TC MESH*, através do TC, permite ao usuário evitar o uso ineficiente da banda, criando diversas classes de serviços. Oferecendo assim diversas prioridades dependendo dos tipos de fluxos.

2.5.4. CONTROLE DE ADMISSÃO DE CHAMADAS (CAC)

O controle de admissão faz-se necessário em duas situações. De acordo com a disponibilidade de recursos na rede, o acesso aos recursos da rede pode ser negado, de modo a balancear o tráfego de entrada na rede e o tráfego encaminhado pelo nó, e no tráfego de *backbone* entre os nós *mesh*, para controle da carga da rede. Basicamente, este mecanismo deve se basear na capacidade disponível da rede, no tipo de tráfego e nos requisitos de QoS do mesmo para determinar se a rede é capaz de aceitar um novo fluxo sem degradar aqueles em curso e garantindo-lhe uma QoS mínima.

2.5.5. QOS FIM-A-FIM

A qualidade dos fluxos fim-a-fim depende da capacidade de negociação e adaptação dos níveis de serviço oferecidos pelas diferentes redes ou domínios administrativos através dos quais um fluxo trafega. Uma rede pode selecionar, trocar ou modificar seu mecanismo de gerenciamento de recursos para implementar seu nível de serviço com as redes vizinhas de forma independente. A qualidade de serviço fim-a-fim será garantida com os níveis de serviço sendo atendidos em cada rede. A ferramenta *TC MESH* pode ser utilizada para configurar parâmetros de QoS em cada rede separadamente.

2.5.6. MECANISMOS DE NEGOCIAÇÃO DE NÍVEIS DE SERVIÇO (SERVICE LEVEL AGREEMENT – SLA)

Em redes *DiffServ* [RFC 3317, 2009] cabeadas, muitas vezes existe a figura de um *Bandwidth Broker* que centraliza as políticas de QoS do domínio, as aplica nos nós e negocia os níveis de serviço com outros domínios. Nas redes *mesh*, os mecanismos que regem a rede são essencialmente distribuídos, introduzindo um novo desafio para esta negociação de níveis de serviço entre domínios diferentes. Em [ACHARYA et al, 2002] é proposto uma arquitetura para provisão de QoS em redes sem fio, não especificamente para redes *mesh*. A arquitetura contempla uma entidade centralizadora (QGS) semelhante a um *Bandwidth Broker* para negociação de QoS com as fontes de tráfego e com estas mesmas entidades de outros domínios e para atualização das tabelas de entidades locais (QLN) responsáveis pela aplicação da política de QoS e pelo transporte do tráfego. Como a atualização é feita para todas as entidades locais do domínio administrativo, a estação tem flexibilidade de mobilidade, mantendo suas garantias de QoS. Para negociação de QoS e controle de admissão (CAC), é utilizado o protocolo *Dynamic Service Negotiation Protocol* (DSNP). Para o

provimento de níveis de serviços diferenciados no nível de rede é importante observar quais são as limitações da rede em termos de vazão pelo número de saltos para especificar quais valores são possíveis oferecer.

2.5.7. DIFERENCIAÇÃO ENTRE TRÁFEGO DE ACESSO/TRÁFEGO DE BACKBONE E BALANCEAMENTO DE CARGA

Os nós de uma rede *mesh* podem exercer funções apenas de núcleo de rede ou associar a estas, funções de ponto de acesso. Com isso, existem dois tipos de tráfego concorrentes, que influenciam na capacidade da rede, no controle de fluxo, no gerenciamento das filas e do espaço em buffer e na diferenciação dos serviços: o tráfego de acesso e o de *backbone*.

A implementação de mecanismos de balanceamento de carga distribui melhor a carga na rede, evitando atrasos, descarte de pacotes ou sobrecarga de um determinado nó ou *gateway* [CHEN et al, 2000]. Isso pode ser obtido com protocolos de roteamento com possibilidade de múltiplos caminhos, desde que exista algum mecanismo de evitar que o tráfego de um mesmo fluxo passando por caminhos diferentes concorra entre si reduzindo a eficiência.

2.5.8. ALOCAÇÃO DE POTÊNCIA

No oferecimento de QoS em uma rede em malha sem fio, é importante levar em consideração a possibilidade do aumento da rede por se tratar de uma solução escalar. Este ponto traz a consideração de que o aumento da potência de transmissão e consequentemente do alcance incrementa as perdas de forma quadrática pelo aumento do conflito espacial [ACHARYA et al, 2002]. Deve haver então um compromisso entre a manutenção da conectividade e a redução do conflito espacial, aumentando a capacidade da rede.

2.5.9. AMBIENTES DE MÚLTIPLOS CANAIS

O uso de múltiplas interfaces e múltiplos canais de frequência aumenta a capacidade da rede, na medida em que faz uma melhor distribuição do tráfego e reduz os problemas do “terminal escondido” e do “terminal exposto”. Porém, estes ambientes oferecem uma maior complexidade na estimativa da disponibilidade da rede para cálculo de métricas baseadas em QoS e para a própria implementação de uma arquitetura voltada para a qualidade de serviço [GERK, 2005].

3. TRABALHOS RELACIONADOS

Neste capítulo serão descritas algumas ferramentas relacionadas diretamente com qualidade de serviço que têm como objetivo facilitar a manipulação de políticas de QoS. Sejam elas para introduzir uma nova forma de desenvolvimento de políticas ou uma ferramenta de gerência de QoS, que permite a criação das políticas e avaliação do comportamento das mesmas.

3.1. TRAFFIC CONTROL NEXT GENERATION (TCNG)

O controle de tráfego no GNU/Linux contém uma série de funcionalidades. Entretanto, a sintaxe do TC é considerada complicada não somente pela necessidade de conhecimentos técnicos sobre o assunto, mas também pela linguagem bastante singular que é usada no mesmo. A abordagem do TCNG define uma nova sintaxe mais amigável, com uma escrita muito semelhante a da programação C, que inclui um compilador próprio para verificação de erros. Estas facilidades foram incluídas através da inserção de uma outra camada de abstração (tcc) mostrada na Figura 6 onde existem as interfaces de entrada de dados (*User or Application*) que são interligadas pela linguagem TCNG para a ferramenta TC onde a mesma se comunica diretamente com o *kernel* do sistema operacional [TCNG, 2008].

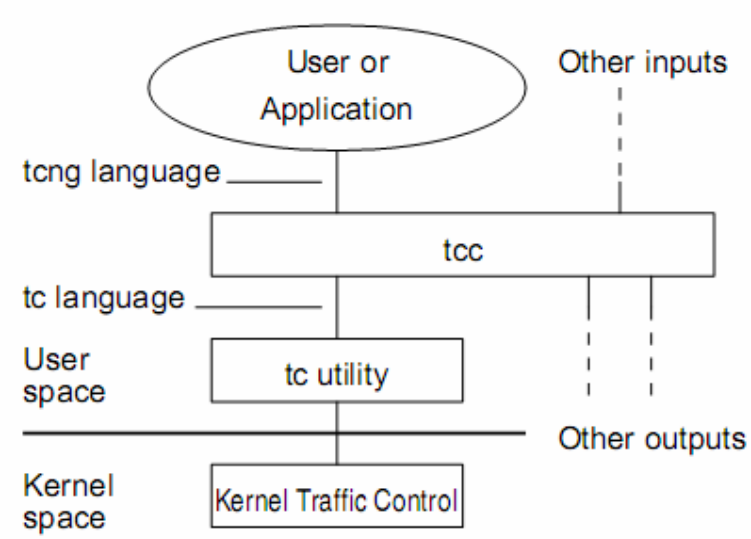

Figura 6. Nova configuração do controle de tráfego no linux [TCNG, 2008].

O compilador de controle de tráfego “tcc” coleta as informações dos scripts com a nova linguagem “tcng” e a traduz para a sintaxe tradicional do TC. Com o TCNG é possível

usar essa nova linguagem sem a necessidade de nenhum tipo de alteração no *kernel* ou no utilitário TC. Adicionalmente, é TCNG permite a adição de "*parsers*" de outras linguagens como o XML.

A linguagem do TCNG é bastante similar ao C e Perl, com a estrutura expressada implicitamente com a sintaxe da linguagem, com variáveis e expressões aritméticas. Os filtros podem ser expressos com a sintaxe da linguagem C, e o compilador "cpp" pode ser usado para incluir arquivos e escrever macros.

A Figura 7 mostra um exemplo de um script TCNG, onde a configuração começa com o nome da interface de rede (eth0, neste caso) e as regras como, por exemplo, os fluxos egressantes e ingressantes. Então, as regras são expressas da seguinte forma:

- ✓ **Ação *if* expressão;**

Onde *ação* pode ser a seleção de uma classe ou o descarte de um tipo específico de pacote. A expressão booleana usa a mesma sintaxe do C. Os cabeçalhos mais comuns como IP, UDP e TCP têm comandos predefinidos, como a expressão "*tcp_dport*" e "*ip_src*" que indicam a porta TCP de destino e o endereço IP de origem, respectivamente.

```
dev "eth0" {
    egress {
        class (<$high>) if tcp_dport == 80;
        class (<$low>) if 1;

        prio {
            $high = class {
                tbf(limit 10kB,rate 20kbps,
                  burst 2kB,mtu 1500B);
            }
            $low = class {
                fifo(limit 30kB);
            }
        }
    }
}
```

Figura 7. Exemplo de script TCNG.

Diferentemente da abordagem inicial do TC, onde os scripts criados eram testados somente no momento da inserção do mesmo na interface de rede, o TCNG permite a simulação do script na ferramenta chamada "*tcsim*" (*Traffic Control Simulator*). Para realizar a simulação, é necessária simplesmente a inserção do código *tcng* na ferramenta de simulação

com algum *framework* para configuração e geração dos gráficos. Isto é mostrado na Figura 8, onde no "*tcsim*" é configurado os parâmetros da ferramenta TCNG e simulado em uma topologia de rede especificada pela ferramenta, onde a partir dos resultados (vazão, pacotes perdidos, etc.) é possível verificar a eficiência do script desenvolvido através dos gráficos (*plot*) gerados.

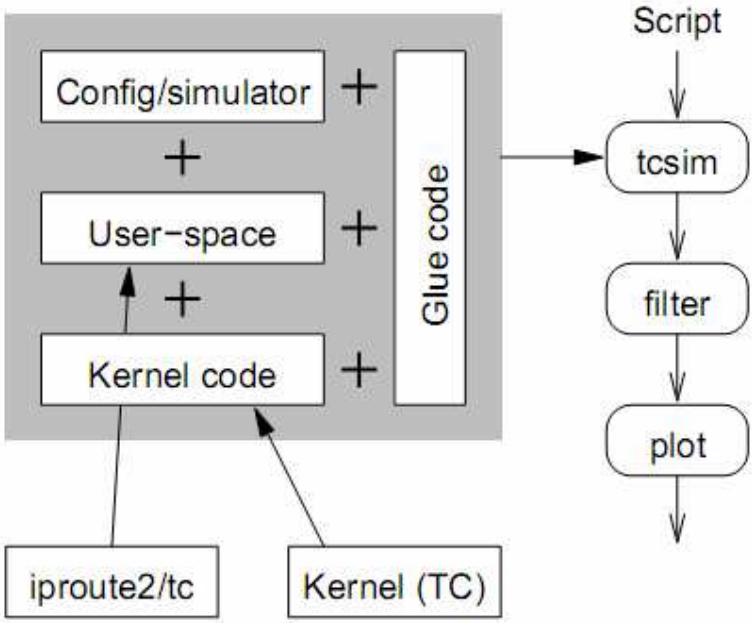

Figura 8. Estrutura do tcsim [TCNG, 2008].

A saída do "tcsim" é simplesmente um gráfico traçado cronologicamente de todos os eventos ocorridos no simulador. Esta saída pode ser filtrada e plotada em um novo gráfico. Concluindo, o TCNG é uma ferramenta versátil que facilitou e ampliou o uso do TC apresentando uma nova forma de configuração mais amigável para os usuários habituados com as sintaxes de programação das linguagens mais utilizadas no meio acadêmico.

3.2. OPENBSD STATEFUL PACKET FILTER

O Packet Filter (*pf*) [HARTMEIER, 2002] é usado como firewall padrão do sistema operacional OpenBSD, entretanto, ele tem funções de filtragem de pacotes e assim também pode ser considerado como uma ferramenta de configuração de QoS.

O *pf* reside no *kernel* do sistema operacional *OpenBSD* e inspeciona qualquer pacote IP que entra ou deixa a interface de rede. Em cada pacote, ele toma uma das seguintes decisões:

- ✓ Autoriza a passagem do pacote, fazendo ou não alguma modificação em seu cabeçalho.
- ✓ Bloqueia silenciosamente o pacote, de acordo com as regras estabelecidas pelo usuário.
- ✓ Rejeita o pacote com uma resposta ao usuário.

3.2.1. REGRAS DE FILTROS

Cada pacote é comparado com regras dos filtros estabelecidos semelhante ao TC. Esta regra consiste em um banco de regras onde cada regra contém uma série de parâmetros que determinam que tipo de ação realizar em um determinado pacote. Este parâmetro pode ser o endereço de origem e/ou destino, protocolo, portas, entre outros.

O *pf* permite não somente que se investigue um pacote isoladamente, mas sim conexões inteiras poderão ter uma regra específica.

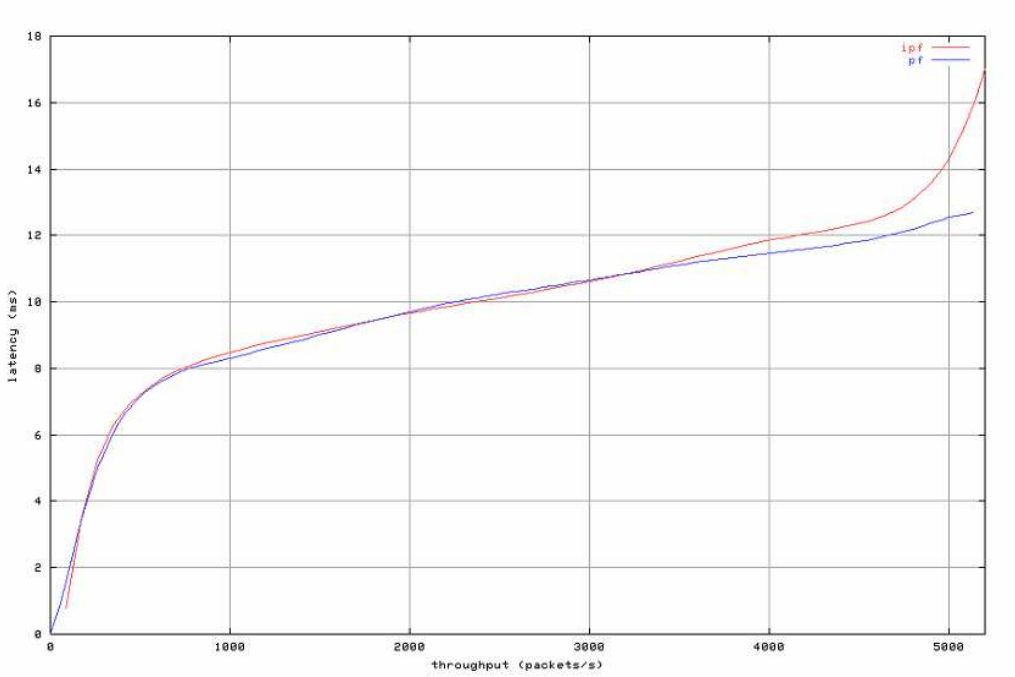

Figura 9. Comparando a eficiência entre as ferramentas ipf e pf [HARTMEIER, 2002].

A Figura 9 mostra uma comparação da eficiência do *pf* em relação ao IPfilter [IPFILTER, 2009], que é um um *software* usando nos sistemas operacionais FreeBSD e NetBSD que oferece serviços de *firewall* e controle de tráfego. O procedimento consistiu no envio de pacotes aumentando a vazão progressivamente para verificar a latência entre os dois métodos citados. É possível perceber uma maior estabilidade do *pf* em relação IPfilter quando os filtros encontram-se sobrecarregados.

3.3. ALTQ - ALTERNATE QUEUEING FOR BSD UNIX

ALTQ [ALTQ, 2009] é um *framework* alternativo para sistemas operacionais baseados em Unix que provê a realização de compartilhamento de recursos e qualidade de serviço. O *framework* oferece componentes de QoS que realizam diferenciação de serviços e adicionalmente oferece ferramentas de gerência que exibem todo tráfego que sai da interface de rede desde que os pacotes passem por alguma fila estabelecida pelo programa. ALTQ está incluído nos sistemas operacionais FreeBSD, NetBSD e é integrado dentro do *packet filter* do OpenBSD.

Com o ALTQ, os pacotes são alocados em filas, como no TC, com o propósito de controlar a banda de uma determinada interface de rede. Em cada fila, é possível o uso de disciplinas de enfileiramentos que vão decidir se os pacotes serão atrasados, descartados ou enviados imediatamente. Atualmente, o ALTQ oferece suporte a três tipos de disciplinas de enfileiramento: *Class Based Queueing* (CBQ), *MultiBand Priority Queueing* (PRIO) e *Hierarchical Fair Service Curve* (HFSC), onde filas são inseridas em uma interface formando uma árvore, e cada fila pode ter filas filhas. Cada fila tem uma prioridade específica e uma banda máxima estipulada.

3.4. JOBS: JOINT BUFFER MANAGEMENT AND SCHEDULING FOR DIFFERENTIATED SERVICES

JoBS (*Joint Buffer Management and Scheduling for Differentiated Services*) [LIEBEHERR e CHRISTIN, 2001] trata-se de um algoritmo de gerência de buffer e marcação de pacote para prover serviços diferenciados baseados em diversas classes. Este algoritmo provê diferenciação independente de cada nó, levando em consideração o atraso e a perda de pacotes para delimitar os fluxos mais prioritários. O algoritmo propõe uma gerência otimizada de buffer e de marcação de pacotes relativa e absoluta em relação aos requisitos que cada classe deseja.

Assume-se que cada enlace de saída permite uma bufferização por classe de fluxo que sai do dispositivo de rede, e que esse tráfego é transmitido a partir dos *buffers* usando um algoritmo com um serviço dinâmico de alocação de taxas de transmissão em cada classe. Tráfegos da mesma classe são transmitidos como no algoritmo "*first-in first-out*". Não há controle de admissão ou policiamento de tráfego. Uma série de requisitos de cada classe é especificada no algoritmo juntamente com as classes em si. Diferentemente do TC, JoBS usa também parâmetros relativos em cada classe. A seguir, temos um exemplo de três classes com seus respectivos parâmetros de QoS.

- ✓ Classe 1 – Atraso 2 vezes maior do que a Classe 2.
- ✓ Classe 2 – Taxa de perdas 1% da taxa da Classe 3.
- ✓ Classe 3 – Atraso menor ou igual a 5 ms.

No exemplo, as duas primeiras classes contêm parâmetros de QoS relativos e a última com parâmetro absoluto. O conjunto de parâmetros pode ser combinado com ambos os tipos.

Para explorar todas as funcionalidades do algoritmo devemos assumir primeiramente que haja recursos computacionais disponíveis para otimizar da melhor maneira possível as classes em cada enlace.

3.5. QOSBOX VISUALIZATION

O QoSbox visualization [CHRISTIN e LIEBEHERR , 2006] é um visualizador gráfico usado em JoBS [LIEBEHERR e CHRISTIN, 2001]. Sua principal contribuição é visualização com gráficos animados dos diferentes comportamentos de um determinado nó da rede suportando diversos tipos de mecanismos de enfileiramento e algoritmos de descarte de pacotes para serviços diferenciados. Nesta ferramenta, o usuário escolhe um mecanismo de enfileiramento e de descarte. Ele então escolhe o destino de um determinado fluxo de dados e o tipo de tratamento de QoS aquele fluxo vai receber.

Uma vez escolhidas as disciplinas de enfileiramento e descarte e o fluxo de dados das simulações, a simulação é iniciada e os gráficos criados com cada classe, atrasos, vazão, entre outros configuráveis pelo usuário.

3.6. QAME (QOS-AWARE MANAGEMENT ENVIRONMENT)

QAME [GRANVILLE et al, 2001] é uma ferramenta de gerência de rede com suporte a QoS baseado no sistema PBNM - *Policy-Based Network Management* [CHANGKUN, 2000], que permite controlar o comportamento de sistemas através da definição de políticas em alto nível. Esta ferramenta usa tecnologias que se acredita serem mais apropriadas para um *backbone* de dimensões nacionais e por isso esse sistema foi criado especificamente para o uso na Rede Nacional de Ensino e Pesquisa (RNP), entretanto, pode também ser usada em outros ambientes como a Internet.

O sistema PBNM tem uma arquitetura composta por quatro componentes: ferramenta de políticas, repositório de políticas, *policy decision point* (PDP) e o *policy enforcement point* (PEP), como exibido na Figura 10.

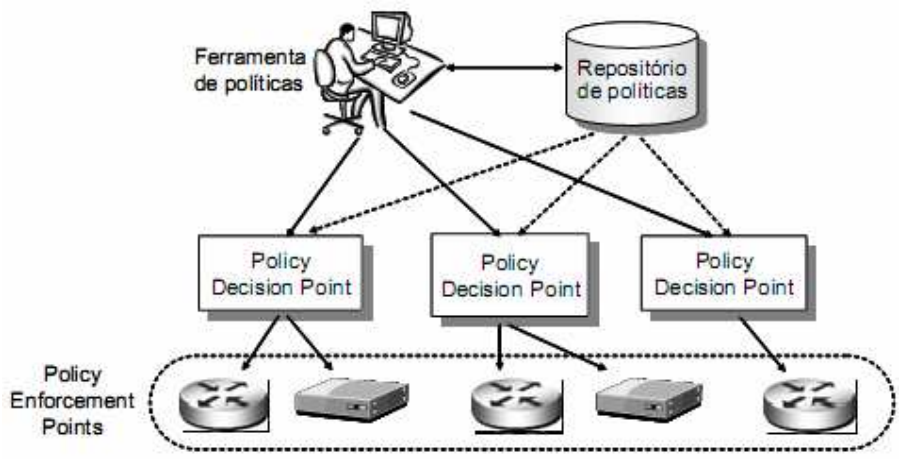

Figura 10. Arquitetura do PBNM [GRANVILLE et al, 2001].

A partir da ferramenta de políticas o administrador define e edita as políticas de gerenciamento que serão armazenadas no repositório de políticas para uso futuro. Para que uma política seja aplicada, a ferramenta sinaliza os PDPs, indicando que eles devem consultar, junto ao repositório de políticas, as informações referentes à política em questão. Cada PDP fará então uma tradução da política para os comandos de configuração que serão aplicados no PEP (ex.: interfaces de rede, disciplinas de fila, etc.). Como repositório de políticas, usou-se o LDAP (*Lightweight Directory Access Protocol*).

- ✓ **Ferramenta de políticas**

A ferramenta de políticas é utilizada pelo administrador para solicitar as ações que serão enumeradas a seguir (vide Figura 11):

Edição de políticas. Possibilita a criação, a modificação e a remoção de políticas. O módulo realiza as operações acessando o repositório de políticas LDAP externo.

Cadastro de PEPs. Os dispositivos alvos, assim como seus PEPs, precisam ser registrados no sistema para serem gerenciados.

Cadastro de PDPs. No sistema implementado, PDPs são PCs rodando Web Services, que proveem a comunicação entre a ferramenta de políticas e os PDPs, mesmo que o ambiente entre eles seja a Internet . Esses dispositivos também necessitam ser registrados junto à ferramenta de políticas, a fim de serem usados no processo de aplicação de políticas.

Associação entre PDP e PEP. Cada PEP deve ser controlado por um único PDP enquanto um mesmo PDP é capaz de controlar diversos PEPs. Cada associação PDP/PEP precisa ser igualmente registrada na ferramenta de políticas. Isso permite à ferramenta selecionar o PDP apropriado para aplicar uma política em um PEP.

Aplicação de políticas. Para aplicar uma política, o administrador seleciona a política desejada e o PEP alvo. Então, a ferramenta entrega a política para o PDP apropriado. A ação de aplicar uma política também cria uma associação entre a mesma e o PEP alvo.

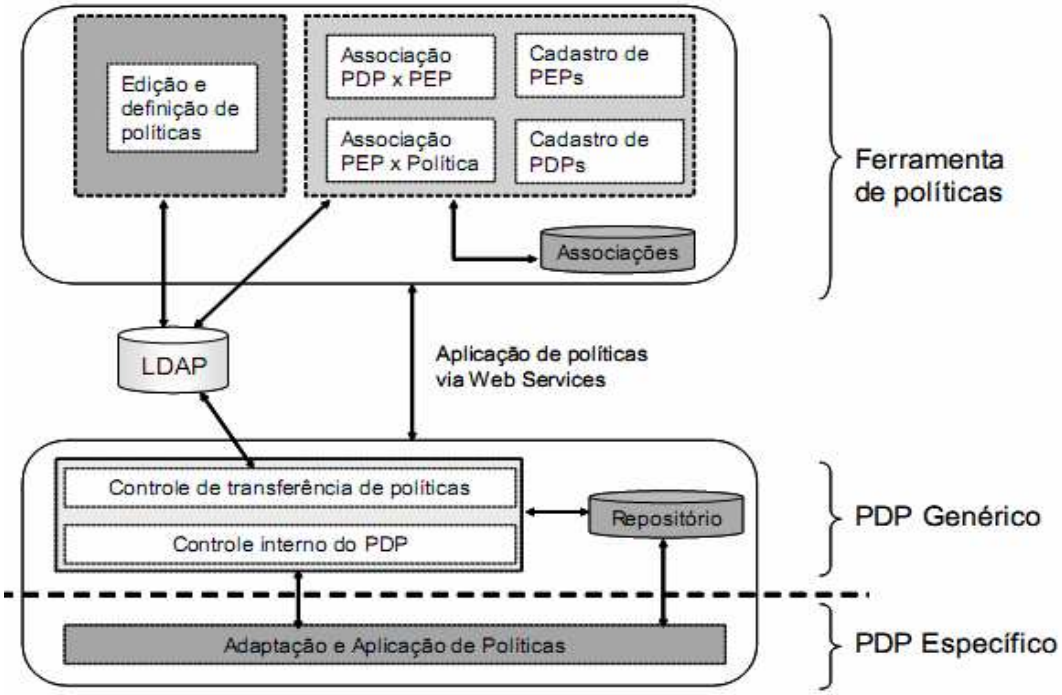

Figura 11. Arquitetura detalhada do QAME [GERK, 2005].

- ✓ **Interface com o usuário**

A ferramenta para políticas foi desenvolvida como um módulo do ambiente QAME. Esse ambiente é constituído por mapas de redes que apresentam os dispositivos que formam o segmento de rede em questão (Figura 12). Inicialmente é preciso configurar o ambiente para suportar os elementos do sistema PBNM, ou seja, é preciso definir os dispositivos que vão atuar como PDPs e PEPs, e cadastrar as políticas que poderão ser aplicadas nos dispositivos. Somente após essas etapas de configuração e definição é que uma política pode ser aplicada nos dispositivos do ambiente QAME.

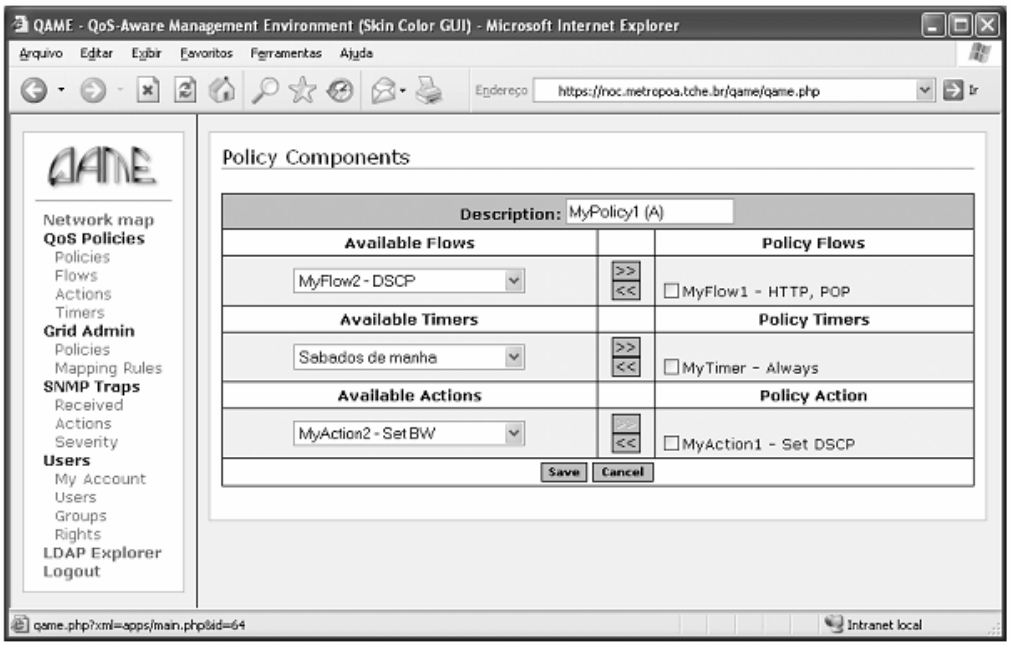

Figura 12. Interface para o usuário do QAME [GRANVILLE et al, 2001].

Após a criação de uma política, o usuário pode visualizar como a tradução da mesma será feita em um PDP específico. Essa característica ajuda o operador a verificar se a configuração final, a ser aplicada em um dispositivo alvo, está adequada.

3.7. PHPTCADMIN

A ferramenta *TC MESH* usa os mesmos conceitos principais da ferramenta phpTCadmin [CARVALHO e ABELÉM, 2008], que é um software que permite a criação de políticas de QoS implementadas na rede através de uma interface web, facilitando a criação de disciplinas de enfileiramento, classes e classificadores em estruturas complexas. O

phpTCadmin possui dois componentes principais: um que interage com a API do controle de tráfego (TC) do linux através do utilitário TC, e outro que exibe de forma intuitiva para o usuário as disciplinas de enfileiramento, classes e classificadores. O phpTCadmin permite a definição de disciplinas de enfileiramento, classes e classificadores em tempo de execução e sem a necessidade de reaplicação das regras existentes. O phpTCadmin manipula todos os algoritmos disponíveis na API do TC do kernel Linux: *Class Based Queuing* (CBQ), *Hierarchical Token Bucket* (HTB), *Hierarchical Fair Service Curve* (HFSC), *MultiBand Priority Queueing* (PRIO), *Stochastic Fairness Queueing* (SFQ), *Token Bucket Filter* (TBF), *Random Early Detection* (RED), assim como suporte aos classificadores U32 e Netfilter Mark, possuindo suporte inicial aos algoritmos *Generic Random Early Detection* e *Diffserv Mark*. Uma limitação da ferramenta phpTCadmin é a seu suporte limitado a definição de filtros e o suporte parcial ao *Diffserv*. Além disso, o phpTCadmin oferece a criação de políticas específicas para desktops e laptops, sendo necessária a inclusão de configurações específicas para o uso em roteadores.

Apesar de todas as funcionalidades citadas, o phpTCadmin apresenta algumas limitações. Primeiramente, é uma ferramenta desenvolvida para uso local, ou seja, não existe nenhum tipo de mecanismo que possibilite a instalação das políticas em um dispositivo remoto, pois a ferramenta desenvolve scripts baseados nas configurações do dipositivo na qual ela está instalada (interface de rede e visualizador das etapas de desenvolvimento do script, por exemplo). Outro característica da ferramenta é a necessidade de que o usuário tenha conhecimentos intermediários sobre o TC para utilizar a ferramenta, pois precisa indicar rigorosamente todos os parâmetros possíveis para o desenvolvimento de um script. Normalmente, tais parâmetros são conhecidos apenas por usuários que já possuem experiência no desenvolvimento de scripts TC.

Outra limitação da ferramenta é a impossibilidade da visualização dos parâmetros de QoS aplicados nos dispositivos de rede, dificultando o trabalho do administrador em analisar a eficiência das políticas aplicadas.

Além dos fatores citados, existe um problema de segurança com a ferramenta, que permite que usuários comuns tenham acesso a privilégios de um superusuário no sistema operacional GNU/Linux.

3.8. RRDTOOL

O RRDTool [RRDTOOL, 2009] (*Round Robin Database*) é um sistema de base de dados desenvolvido para permitir o armazenamento de séries de dados numéricos sobre o

estado de redes de computadores. Entretanto, ele pode ser usado no armazenamento de qualquer outra série de dados como temperatura, uso de CPU, etc. A base de dados desta ferramenta possui um tamanho máximo que não pode ser ultrapassado. Os dados armazenados são consolidados de acordo com a configuração fornecida, de forma que a resolução deles seja reduzida de acordo com o tempo que os dados estão armazenados. Neste processo, somente as médias dos valores antigos são armazenados. Adicionalmente, o RRDtool produz gráficos que oferecem ao usuário a visualização gráfica dos dados armazenados.

O RRDtool pode ser intergrado a outras ferramentas para ampliar a sua utilização. A ferramenta Cacti [CACTI, 2009] e MRTG (*Multi Router Traffic Grapher*) [MRTG, 2009] são exemplos de ferramentas que, em conjunto com o RRDtool, oferecem opções como a monitoração em tempo real da rede através do protocolo SNMP [RFC 1157, 2009].

4. CONTROLE DE TRÁFEGO NO LINUX

O controle de tráfego através da ferramenta *traffic control* (TC) do linux funciona através de algoritmos que manipulam como os dados são transmitidos para as respectivas interfaces de rede. Esses algoritmos são responsáveis em enfileirar, priorizar, descartar e oferecer diferenciação de serviços a um determinado tráfego. A utilização de controle de tráfego permite solucionar problemas de gerência de redes limitando, por exemplo, a taxa de transmissão de pacotes de um PC ou qualquer dispositivo que contenha uma distribuição linux com o linux *traffic control* instalado. O TC oferece adicionalmente mecanismos de qualidade de serviço como o *Diffserv* e *Intserv*, este último através do protocolo RSVP [RFC 2205, 2009].

O subsistema de rede do *kernel* linux necessita definir quais são os pacotes que serão tratados, analisando qual ação executar em cada pacote. O *kernel* executa esta tarefa de acordo com o algoritmo de escalonamento de pacotes escolhido. O algoritmo padrão para o escalonador de pacotes é FIFO (*First In First Out*), onde o primeiro pacote a chegar é o primeiro a ser transmitido pelo *driver* da interface de rede.

O *kernel* é estruturado utilizando os conceitos de disciplinas de enfileiramento (*qdiscs - queue disciplines*), classes (*class*) e classificadores que também são chamados de filtros (*filters*). Através destes componentes é possível moldar fluxos de dados das interfaces de rede. Dessa forma, vários algoritmos podem ser implementados para serem associados a estas estruturas. A Figura 13 exemplifica como estes componentes são relacionados.

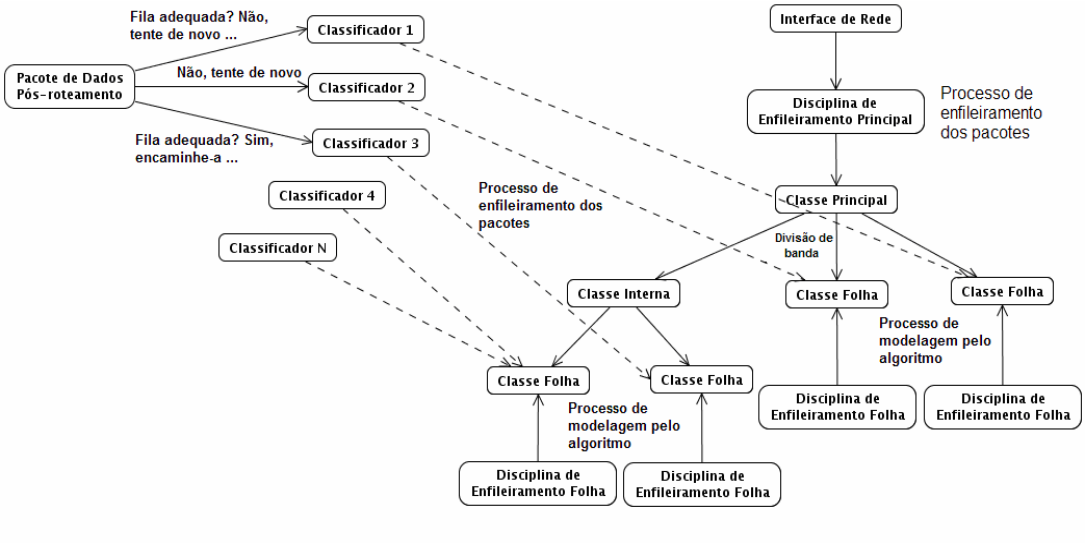

Figura 13. Estrutura de implantação de classes com *traffic control*.

Cada interface de rede no qual se deseja implantar políticas de QoS necessita de uma disciplina de enfileiramento principal. Nesta disciplina, poderá haver a possibilidade de se incluir subclasses (*classful*) ou não (*classless*). Na Figura 13, temos um exemplo de disciplinas *classful*, sendo criada uma classe principal seguida de uma classe secundária (chamada também de classe interna). Uma classe interna é uma classe subordinada a uma classe principal e pode ter classes folhas. Por último temos as classes folhas (também chamadas de subclasses), ou seja, classes terminais. Em cada classe folha, é possível escolher uma disciplina de enfileiramento específica que melhor atenda as necessidades do administrador da rede. Cada pacote é encaminhado para as classes de acordo com as informações dos classificadores. Tais classificadores agem sobre informações dos cabeçalhos de nível de rede e transporte da arquitetura TCP/IP. Cada interface de rede pode usar sua própria disciplina de enfileiramento independente de outra. O tratamento do tráfego nas interfaces deve ser realizado por um nó que esteja interligando redes

4.1. VISÃO GERAL DA ESCRITA DE UM SCRIPT TC

Para o desenvolvimento de um script com definições de políticas de QoS são necessários basicamente três componentes: disciplina(s) de enfileiramento (*qdisc*), classes (*class*) e classificadores (*filter*). A partir da definição destes componentes é possível criar diversas políticas para os mais diversos tipos de objetivos, sejam eles visando aplicações multimídia ou outra modalidade de tráfego [HUBERT, 2009].

- ✓ **Sintaxe**

A sintaxe para gerenciar uma disciplina de enfileiramento é:

tc qdisc [add | del | replace | change | get] dev STRING
[handle QHANDLE] [root | ingress | parent CLASSID]
[qdisc]

Onde:

tc

É o próprio utilitário.

qdisc

É a disciplina de enfileiramento atual { [p|b]fifo | tbf | prio | htb | etc. }

add

Adiciona uma disciplina de enfileiramento em uma interface de rede.

del

Remove uma disciplina de enfileiramento em uma interface de rede.

replace

Substitui uma disciplina de enfileiramento por outra.

handle

É a identificação de uma determinada disciplina de enfileiramento. Cada disciplina tem um número que a representa, este número é chamado de *handle*.

root

Indica que a disciplina de enfileiramento está na raiz do script, ou seja, é a qdisc principal. Esta disciplina é a responsável por distribuir toda a banda para todas as subclasses. Existe somente uma qdisc por interface.

ingress

Policia o ingresso dos pacotes.

parent

Indica que uma classe faz parte de uma determinada qdisc raiz.

dev

É a interface de rede que está anexada a disciplina de enfileiramento.

Cada interface de rede tem uma disciplina de fila associada a ela que controla como os pacotes são enfileirados e tratados. É possível obter essas informações com o seguinte comando:

root@bruwand:/home/bruno# ip link show

1: lo: <LOOPBACK,UP> mtu 3924 qdisc noqueue

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

2: eth0: <BROADCAST,MULTICAST,PROMISC,UP> mtu 1500 qdisc pfifo_fas t qlen 100

link/ether 52:54:00:de:bf:19 brd ff:ff:ff:ff:ff:ff

3: tap0: <BROADCAST,MULTICAST,NOARP> mtu 1500 qdisc noop

link/ether fe:fd:00:00:00:00 brd ff:ff:ff:ff:ff:ff

Para remover todas as políticas de fila usa-se o seguinte comando:

root@bruwand:/home/bruno# tc qdisc del dev <interface> root

4.2. DISCIPLINAS DE ENFILEIRAMENTO

Quando um determinado pacote de dados está em um buffer de uma interface de rede, o mesmo é alocado para uma fila. A cadência de como esses pacotes saem da fila e como são descartados ou atrasados é o principal conceito de disciplina de enfileiramento. Por padrão, a disciplina de fila *pfifo_fast* é adotada pelo dispositivo de rede quando não há nenhum tipo de manipulação pelo usuário. Esta disciplina tem três subfilas (0-2), sendo o pacote alocado de acordo com a sua prioridade (maior prioridade – 0), caso haja algum tipo de marcação no pacote.

4.2.1. FIFO (FIRST IN FIRST OUT)

É uma disciplina de filas básica *classless* onde todos os pacotes são tratados de forma igual dentro de uma única fila. Todos os pacotes saem da fila na mesma ordem que entraram, como ilustrado na Figura 14.

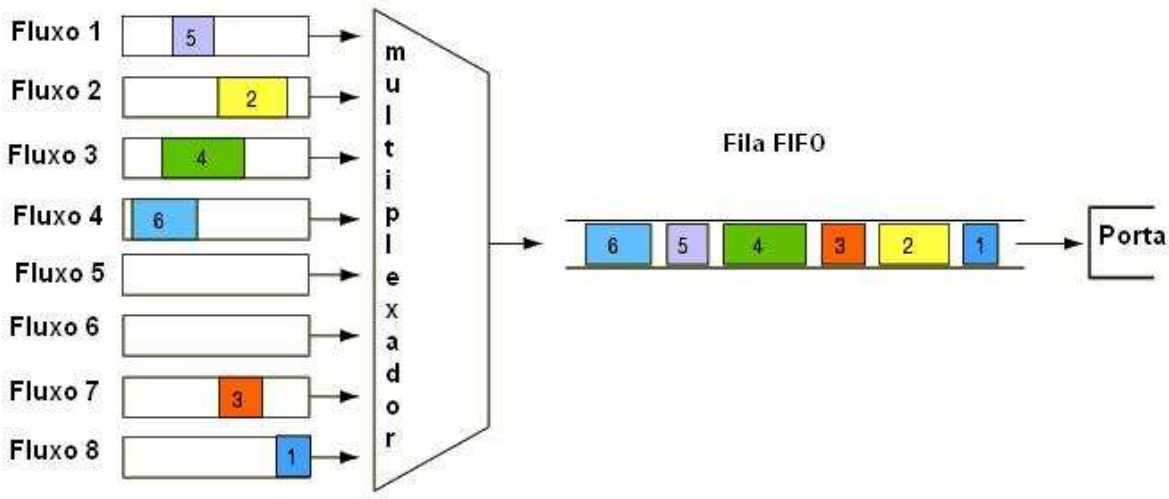

Figura 14. FIFO - todos os fluxos saem na ordem em que entraram [OPALSOFT, 2009] (adaptado).

4.2.2. MULTIBAND PRIORITY QUEUEING (PRIO)

A modalidade de enfileiramento PRIO permite uma diferenciação básica dos serviços. Os pacotes são classificados pelo sistema de filtros para sua respectiva fila, cada uma com prioridades diferentes. A PRIO cria automaticamente três filas secundárias FIFO, onde os pacotes são encaminhados de acordo com suas prioridades (prioridade 1 é maior e 3 é a menor) como na Figura 16. A PRIO pode ser considerada uma disciplina de enfileiramento *classful*, pois contém um número (mesmo que fixo) de filas secundárias. A Figura 16 mostra o funcionamento deste tipo de enfileiramento, onde há diversos fluxos de dados chegando pela interface de rede e redirecionados de acordo com a prioridade que lhes foi dada [HUBERT, 2009].

É possível oferecer prioridade para alguns tipos de fluxo na rede em malha, como por exemplo todos os pacotes de dados multimídia que utilizem o protocolo *Real Time Protocol* (RTP) e aos pacotes de controle do protocolo de roteamento para facilitar a sinalização entre os roteadores. Por último, oferecemos prioridade mínima para os demais pacotes. Na figura 17, temos a disposição lógica das filas.

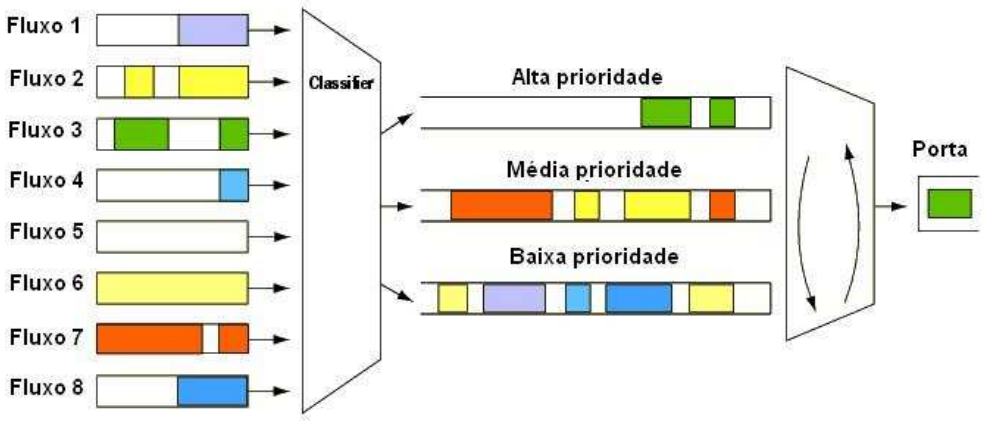

Figura 15. Funcionamento do PRIO [OPALSOFT, 2009] (adaptado).

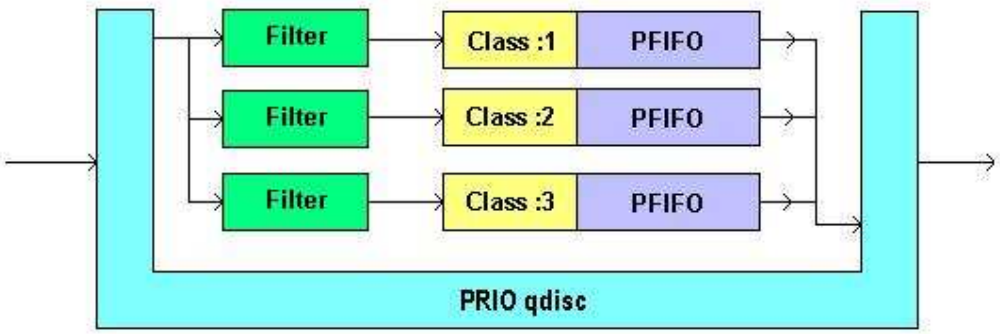

Figura 16. Exemplo de uma disciplina de enfileiramento PRIO [OPALSOFT, 2009].

As principais vantagens da PRIO são a sua carga computacional baixa, quando comparada com outros algoritmos *classful* e permitir também uma melhor organização dos *buffers* dos dispositivos de rede, possibilitando a classificação prioritária de fluxos em tempo real em relação a outros fluxos [HUBERT, 2009].

Há também algumas ressalvas a fazer. Se o fluxo mais prioritário gerar muito tráfego e não for devidamente policiado, ou seja, estabelecidos limites de uso dos recursos da interface de rede, os fluxos classificados com prioridade mais baixa podem sofrer atraso excessivo ou até mesmo serem descartados. Se isto ocorrer, é possível que uma combinação de alta perda de pacotes e aumento da latência em todas as filas prejudique o desempenho do classificador. Consequentemente, o provimento dos três níveis de serviço fica comprometido. Esta política de filas não é a mais indicada para resolver a limitação do FIFO quando um fluxo UDP é favorecido em relação ao TCP em momentos de congestionamento. Se o TCP for alocado em

filas prioritárias, o gerenciador de janela deslizante do TCP juntamente com o mecanismo de controle de fluxo irá consumir toda banda disponível e assim deixará as filas menos prioritárias com UDP sem banda alguma.

4.2.3. TOKEN BUCKET FILTER (TBF)

O filtro *token bucket* é uma disciplina de enfileiramento *classless*, de alta precisão e com baixa necessidade de processamento, onde os pacotes somente saem da fila enquanto não passarem de certa taxa pré-estabelecida, com a possibilidade de ter pequenas rajadas definidas pelo usuário [HUBERT, 2009]. A implementação do TBF consiste em um balde (*bucket*) que contém uma série de "permissões" (*tokens*), permitindo ou não que um pacote ou um conjunto deles saiam da interface de rede. Essa cadência de *tokens* é o que define a vazão da fila, quanto mais *tokens*, maior a rajada permitida.

A principal característica do TBF é a possibilidade da moldagem do tráfego da rede, ou seja, quando o número de *bytes* for maior do que a quantidade de *tokens* disponíveis não será possível que todos os pacotes saiam da interface quando desejam, provocando atrasos e descartes de pacotes.

Os parâmetros necessários para a criação de um script com TBF são:

Limit

Limit é o numero de bytes que podem ser enfileirados aguardando até que *tokens* fiquem disponíveis.

Latency

No TBF podemos especificar o tempo máximo que um pacote pode ficar na fila. Excedido esse tempo, o pacote é descartado.

Burst

Tamanho do balde em *bytes*.

Mpu

O MPU (*Minimal Packet Unit*) determina a quantidade mínima de *tokens* por pacote.

Rate

Como o nome já diz, especifica a taxa (vazão) da fila.

Peakrate

O TBF permite algumas rajadas em curtos espaços de tempo, definidos pelo parâmetro *peakrate*.

Minburst/MTU

É possível definir um tamanho mínimo de pacote que pode passar pela fila, este parâmetro pode até mesmo ser usado como uma espécie de filtro.

Um exemplo é ilustrado pela Figura 17, onde temos uma disciplina de enfileiramento principal (PRIO) e desejamos que em uma dessas filas o TBF seja usado para filtrar um determinado tipo de pacote.

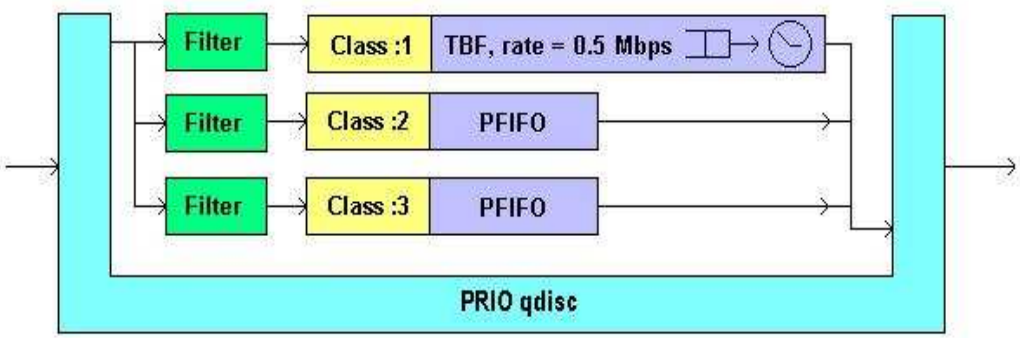

Figura 17. Funcionamento do TBF em uma disciplina classful [OPALSOFT, 2009].

Pelo seu alto grau de precisão e a possibilidade de controle da latência, o TBF pode ser usado em aplicações multimídia como em [MCKENNEY, 1990], onde é estiamado um valor de *token* para estabelecer uma taxa (vazão) efetiva de uma determinada classe onde trafegam vídeos baseados na suíte de protocolos H.323 utilizando a abordagem de serviços integrados.

4.2.4. STOCASTIC FAIR QUEUEING

A disciplina de filas *Stocastic Fair Queueing* (SFQ) é uma disciplina *classless*, derivada dos algoritmos *fair queueing* [HUBERT, 2009]. O SFQ é uma simples implementação do *fair queueing*, funcionando como se houvesse várias portas, ou seja, o tráfego é dividido como em várias filas FIFO, uma para cada fluxo de dados. Essas filas fazem parte de um algoritmo de *round robin*, oferecendo a cada fluxo uma oportunidade igual de envio dos pacotes. Este tipo de fila é chamado de estocástica, pois ela não necessariamente aloca uma fila para cada fluxo, e sim divide o tráfego em um número limitado de filas usando um algoritmo de *hashing*. Por causa deste *hash*, vários fluxos podem ser alocados na mesma fila, ocasionando uma falta de justiça perante algumas classes. Para evitar essa situação, o SFQ mudou esse algoritmo de *hashing* para melhorar a justiça para cada fluxo. Este tipo de

disciplina de filas é muito útil quando sua interface de rede opera a maior parte do tempo na capacidade máxima.

O SFQ contém apenas dois parâmetros específicos:

Perturb

Este parâmetro define quando o hashing é reconfigurado.

Quantum

Especifica a quantidade de bytes (MTU) permitida que passe pela fila. Não é recomendável alterar para valores abaixo de uma MTU padrão ethernet, com risco de grande descarte de pacotes.

Um exemplo de implementação do SFQ é concatenar seu uso com a política de filas PRIO como disciplina de enfileiramento principal. Sabendo que este tipo de política cria também 3 filas secundárias, é possível construir uma política de QoS como na Figura 18.

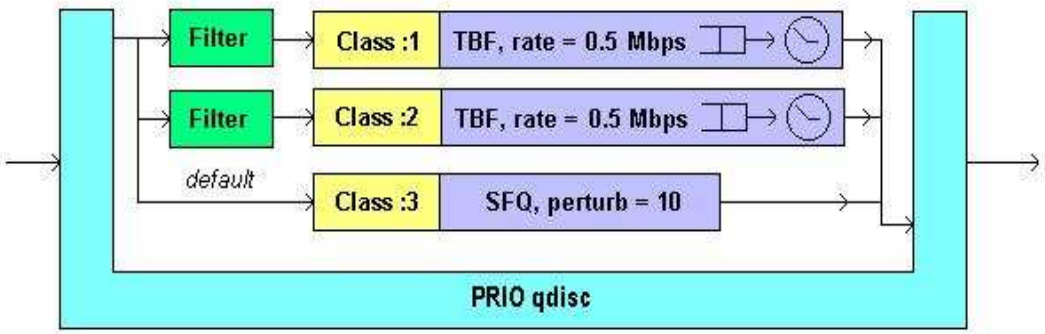

Figura 18. Exemplo do uso do SFQ em conjunto com PRIO e TBF [OPALSOFT, 2009].

4.2.5. HIERARQUICAL TOKEN BUCKET

O HTB é uma disciplina de enfileiramento *classful* baseada em classes hierárquicas onde existem basicamente três tipos de classes: raiz (*root*), filhas (*inner*) e folhas (*leaf*). A classe raiz se encontra no topo da hierarquia e todo tráfego passa através dela [HUBERT, 2009]. Classes filhas são intermediárias, ou seja, fazem parte de uma classe raiz e podem ter classes folhas. Já as classes folhas são as últimas na hierarquia, ou seja, são classes terminais. Nas classes folha, o tráfego das classes de hierarquia mais altas é injetado de acordo com a classificação que foi estabelecida pelos filtros, desta forma torna-se possível a classificação de diferentes tipos de tráfego e prioridades, podendo ter tratamento diferenciado. Antes de o

tráfego entrar nas classes folhas, é necessário que ele seja classificado através de filtros com diferentes regras. Além disso, quando o tráfego é classificado, há a possibilidade de estabelecer uma vazão específica para cada classe. Com o objetivo de executar todas essas tarefas, o HTB usa o conceito de *tokens* e *buckets* para o controle de banda do enlace. Para ajustar a vazão, o HTB gera *tokens* com a cadência necessária para cada banda específica. Os pacotes somente serão encaminhados para fora do buffer quando houver *tokens* disponíveis. Há também o conceito de *borrowing* (empréstimo) [HUBERT, 2009]. Assim, o HTB assegura que a quantidade de serviço fornecida a cada classe é, no mínimo, a quantidade que esta pede, e no máximo o teto. Quando uma classe usa menos do que a quantidade a que tem direito, a largura de banda em excesso é distribuída pelas outras classes que fazem uso de largura de banda. O HTB já está incluído no *kernel linux* a partir da versão 2.4.20. A Figura 19 apresenta a idéia principal.

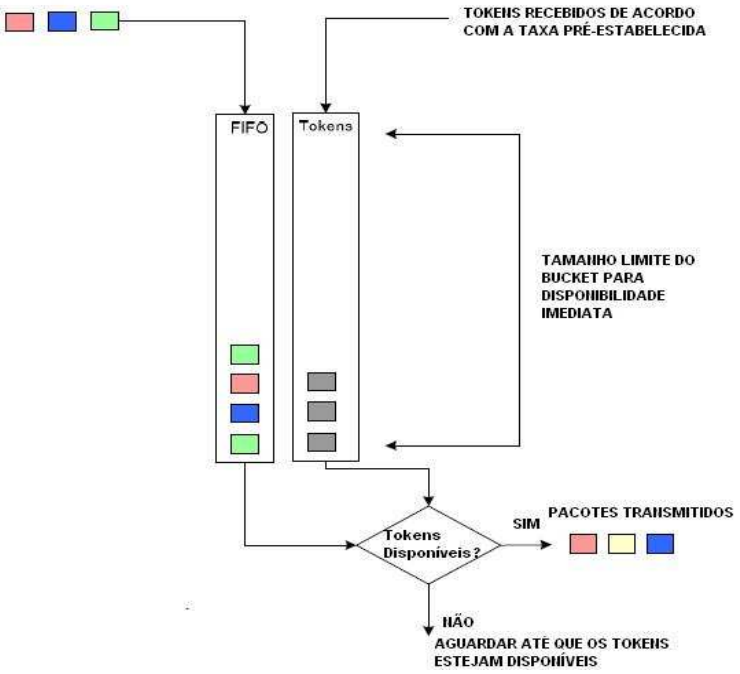

Figura 19. Funcionamento do HTB.

Podemos ter uma hierarquia de classes "*flat*", ou seja, uma certa largura de banda é alocada para cada classe (como na Figura 20, exibindo cada banda alocada em %). Esta alocação de recursos pode ser dinâmica (variando a resposta de acordo com o comportamento da rede) ou estática (permanentemente ajustada pelo administrador da rede). Para as classes que não têm uma porcentagem de banda garantida há a possibilidade de ficar sem transmitir nada em momentos de congestionamento [HUBERT, 2009].

O HTB garante que o total de serviço provido para cada classe é no mínimo o total requerido por esta e conforme o que foi designado para cada classe. Quando uma classe requisita menos que o total a ela reservado, a largura de banda remanescente é distribuída para outras classes que requisitaram serviços.

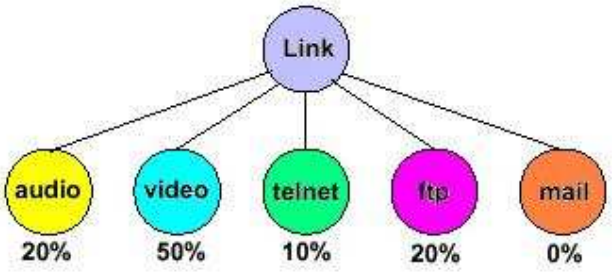

Figura 20. Divisão da banda na interface de rede com hierarquia "flat" [OPALSOFT, 2009].

O segundo tipo segue a abordagem onde a estrutura de compartilhamento do enlace é hierárquica. A Figura 21 mostra o compartilhamento do enlace entre organizações, entre tipos de protocolos, entre classes de serviço e entre conexões individuais sem uma classe de serviço. Todos os pacotes que chegam da interface de rede são encaminhados para cada uma das classes secundárias; as classes folhas são usadas para direcionar de forma mais específica para onde a largura de banda disponível deve ser alocada. A vantagem de implementar uma estrutura hierárquica é a possibilidade de alocar banda para as subclasses que realmente necessitam ou que tenham uma maior prioridade.

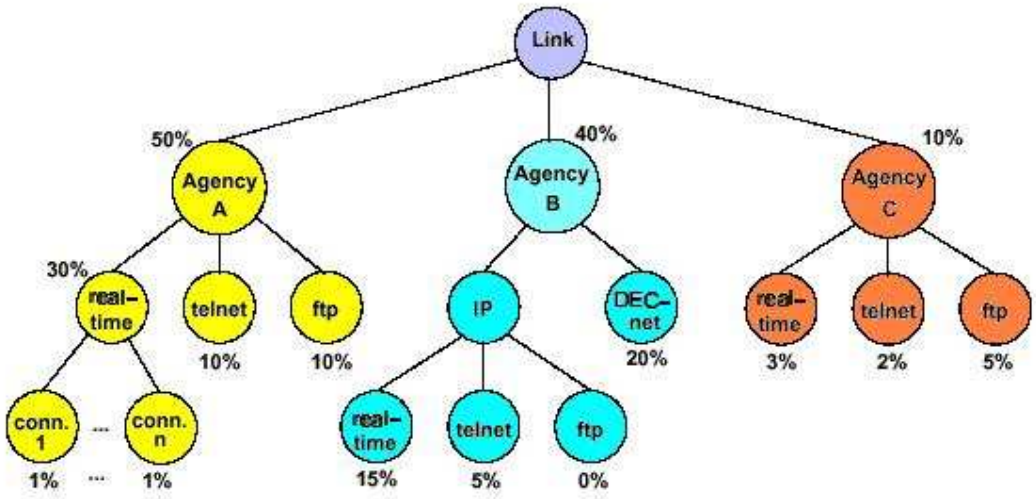

Figura 21. Divisão da banda na interface de rede com hierarquia entre organizações [OPALSOFT, 2009].

Podemos resumir as vantagens de compartilhamento de enlace da seguinte forma:

- ✓ O interior de cada classe secundária pode receber e alocar banda em intervalos de tempo apropriados, com demanda suficiente.
- ✓ Se cada classe folha da organização já estiver com banda suficiente para suas operações, é possível distribuir o excesso de forma homogênea a todas as classes.

Observe que a segunda abordagem (hierárquica) é mais flexível que a "flat". Na abordagem "*flat*" temos apenas um nível de distribuição de banda. Já na abordagem hierárquica é possível obter vários níveis de distribuição de banda aumentando a flexibilidade para distribuir a banda disponibilizada entre as classes.

Devido à flexibilidade do HTB, o mecanismo é utilizado para diversas finalidades. Em [CHUN et al, 2003] é oferecida uma proposta de rede aberta e escalável, que permite à comunidade acadêmica desenvolver novos serviços para a Internet que poderão operar simultaneamente em múltiplos computadores ao redor do mundo. Nessa abordagem, foi usado o HTB para limitar os recursos de cada nó da rede, permitindo assim uma utilização mais homogênea acarretando em um melhor uso dos recursos da rede.

4.2.6. DIFFSERV MARK (DSMARK)

O DSMARK [HUBERT, 2009] é uma disciplina de enfileiramento desenvolvida com as especificações do *Diffserv*. Ela é capaz de marcar o campo *DS Field* do cabeçalho IP, entretanto, ela não modela, controla, descarta, ou prioriza tráfego. A função do DSMARK é apenas marcar o campo *DS Field* [RFC 2474, 2009]. A Figura 22 mostra a definição do campo TOS do cabeçalho IP que informa a marcação do pacote sendo utilizados somente os seis primeiros bits do campo.

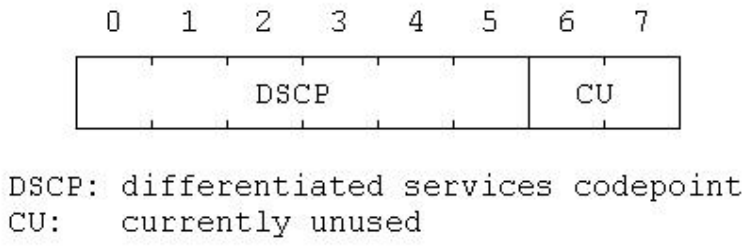

Figura 22. Campo DS-Field no cabeçalho de um datagrama IP [RFC 2474, 2009].

É possível criar "*n*" classes com filtros específicos, onde cada classe irá marcar o pacote com uma combinação de *bits* estabelecida no padrão *Diffserv*. Este padrão é orientado

a grupo, ou seja, o *Diffserv* não conhece detalhes sobre cada fluxo de dados (como propõe o *Intserv*), e sim sobre agregados de fluxos nos quais poderemos aplicar diferentes comportamentos para cada agregado na rede dependendo das características que se deseja.

Quando um pacote chega a um nó com características *Diffserv* é preciso marcar seu campo *DS FIELD* e encaminhá-lo. Os roteadores de borda irão marcá-lo e os nós centrais apenas encaminharão os pacotes de acordo com o comportamento que foi indicado no seu cabeçalho seguindo a classe especificada, como ilustrado na Figura 23.

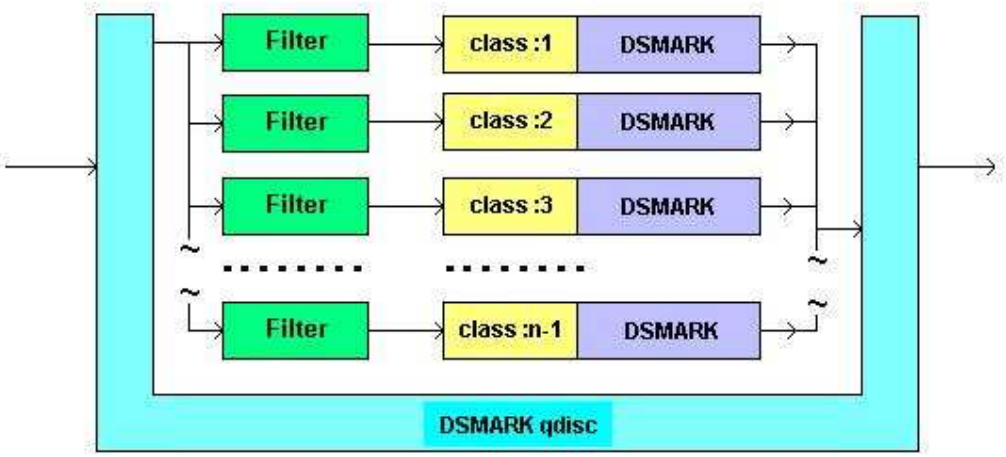

Figura 23. Estrutura de uma disciplina de enfileiramento com DSMARK [OPALSOFT, 2009].

A principal diferença entre o DSMARK e o HTB é basicamente o comportamento das classes onde o HTB apenas controla a vazão que passa sobre ela, diferentemente do DSMARK que apenas marca os pacotes de um determinado fluxo.

Para criar uma disciplina de classes DSMARK usa-se a seguinte sintaxe:

tc add qdisc dev eth0 handle <> root dsmark

indices <> [default_index <>] [set_tc_index]

Este comando indica o DSMARK como disciplina de enfileiramento raiz através da interface eth0 com os seguintes parâmetros:

handle

Como já dito anteriormente, indica a numeração da classe.

indices

É a quantidade de classes que teremos.

Default index

Com este parâmetro, indicamos uma fila onde todos os pacotes que passam por ela não são marcados. Este parâmetro é opcional.

Um exemplo para o DSMARK é a criação de um script onde temos sete classes, modificando o campo *DS field* para certo nível de serviço. De acordo com o filtro de cada classe, o pacote é direcionado para uma fila onde o campo *DS Field* do pacote é modificado. Existe a possibilidade de usar máscaras com o objetivo de preservar uma determinada faixa de bits do campo caso, por exemplo, já tenha sido realizada alguma marcação anteriormente e não se deseje que ela seja sobrescrita. Neste caso (vide Figura 24), estabelecemos níveis de serviço diferentes para redes específicas.

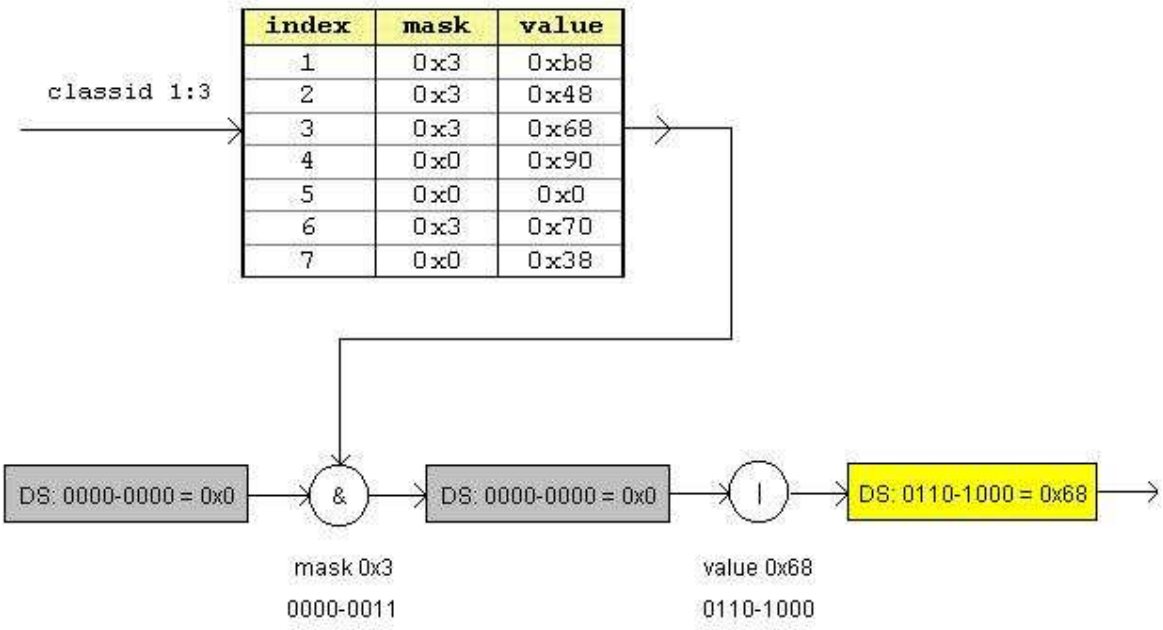

Figura 24. Funcionamento do DSMARK [OPALSOFT, 2009].

4.2.7. RANDOM EARLY DETECTION (RED)

O comportamento normal de uma disciplina de enfileiramento em um roteador é chamado de "*tail drop*". *Tail drops* funcionam enfileirando certa quantidade de pacotes e depois descartando aquilo que não couber mais na fila. Este procedimento não é considerado justo, pois será necessário o uso da retransmissão dos pacotes para correção de erros. Quando uma retransmissão sincronizada ocorre, os pacotes descartados em questão ocasionam um grande fluxo em rajada de retransmissões, congestionando o buffer do roteador mais uma vez. De modo geral, os roteadores de *backbone* frequentemente implementam grandes filas. Essas filas grandes são importantes para um bom desempenho no que diz respeito à vazão,

entretanto, tendem a aumentar a latência e ocasionar mais fluxos em rajada em conexões TCP durante um congestionamento. Esse problema é algo comum na Internet por conta do serviço *best effort* e o aumento de aplicações possíveis na rede. O *kernel linux* oferece o *Random Early Detection*, ou simplesmente RED [FLOYD e JACOBSON, 1993], para atenuar o problema.

O grande trunfo do RED é oferecer um controle de congestionamento pelo controle do tamanho da fila. Outro ponto importante é o descarte aleatório dos pacotes que chegam à fila evitando uma sincronização global e consequentemente distribuindo as perdas entre várias conexões.

O primeiro trabalho do mecanismo de controle de congestionamento é detectar previamente um congestionamento. Este controle mantém a rede em uma região de baixo atraso e alta vazão. O preenchimento médio da fila RED mantém-se em níveis baixos e em alguns momentos ocorre uma flutuação podendo acomodar alguns tráfegos em rajada. Como o roteador monitora o tamanho da fila o tempo inteiro e possui uma visão unificada das várias fontes que contribuem para o congestionamento, ele é o agente que detecta e notifica quando um congestionamento é iminente.

A segunda função do mecanismo de controle de congestionamento é decidir quais conexões serão notificadas do congestionamento. Se o congestionamento for detectado antes do buffer do roteador ficar cheio, não é necessário descartar pacotes para notificar as fontes do mesmo. Segundo [FLOYD e JACOBSON, 1993], o roteador marca o pacote e avisa à fonte para reduzir a janela para aquela conexão. Esta marcação e notificação podem consistir no descarte de um pacote, marcar um bit no cabeçalho do pacote ou outro método conhecido pela camada de transporte ou rede.

Para usar o RED, é necessário especificar três parâmetros: o tamanho mínimo da fila (Min), que consiste em determinar o momento antes dos pacotes começarem a ser descartados, o tamanho máximo de fila (Max), que consiste no momento em que os pacotes começam a ser descartados ou marcados, e finalmente o parâmetro “burst”, que limita o número máximo de pacotes em uma rajada. A Figura 25 mostra o funcionamento básico do algoritmo.

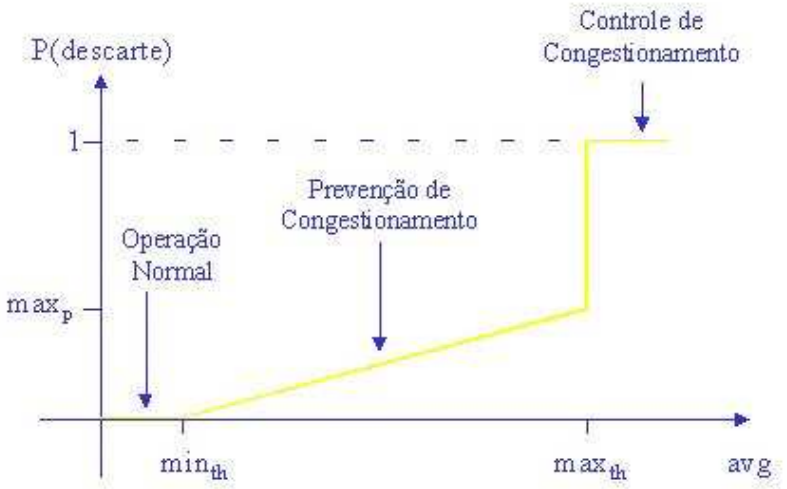

Figura 25. Comportamento da disciplina de enfileiramento RED.

Para o uso no TC, mais alguns parâmetros são necessários:

Probability

É a probabilidade máxima que um pacote tem de ser marcado.

Limit

Tamanho real da fila RED.

Average packet (AVPKT)

Este parâmetro é usado juntamente com o burst para determinar o tamanho da fila para um dado momento.

Banda

Banda a ser disponibilizada na fila.

4.3. FILTROS

Com a ferramenta TC existe a possibilidade de direcionarmos determinados tipos de pacotes para classes e/ou subclasses desejadas [HUBERT, 2009]. É a partir dessa possibilidade que oferecemos vários níveis de prioridade para o pacote, oferecendo mais vazão e/ou menos atraso para uma certa categoria de pacotes. Existem vários tipos de filtros, entretanto, neste texto somente serão explicados os filtros “u32” e alguns filtros “fw”.

4.3.1. FILTROS U32

O filtro *u32* [HUBERT, 2009] é o filtro mais avançado que o TC possui. Ele contém uma lista de gravações que consistem em duas especificações: a seleção e a ação. Esses seletores comparam o cabeçalho IP com o primeiro "*match*" que ocorrer, ou seja, quando houver um parâmetro que esteja de acordo com um determinado filtro esse será identificado e uma ação é executada sobre ele. Um exemplo clássico do uso de ação é o direcionamento para uma classe.

A sintaxe é dividida em três partes: especificação de filtro, seleção e ação, como é mostrado a seguir:

tc filter add dev IF [protocol PROTO]
[(preference|priority) PRIO]

O campo "*protocol*" descreve qual tipo de protocolo será aplicado no filtro. Há outras possibilidades de filtros que serão citados mais adiante. O campo "*preference*" (ou "*priority*") indica a prioridade do filtro em questão. Este parâmetro é relevante desde que existam diversas prioridades e, consequentemente, diversos níveis de serviço. O seletor u32 contém a definição do filtro que será processada no pacote, definindo quais bits serão verificados no cabeçalho.

4.3.2. EXEMPLOS DE FILTROS

A seguir, serão descritos as principais formas de filtragem de fluxos específicos de dados para prover diferentes tipos de prioridades a partir das classes e subclasses criadas pelo TC.

4.3.2.1. FILTRAGEM A PARTIR DA PORTA

Esta modalidade de filtragem permite a filtragem de acordo com a porta de origem ou destino. Desta forma podemos selecionar alguns protocolos de aplicação específicos como o HTTP (porta 80) ou um protocolo de roteamento específico como BGP (porta 179). No exemplo a seguir temos um trecho de um script TC onde todo o tráfego é filtrado originado da porta 20 (*match ip sport 20*), que é do protocolo FTP, para a classe 1:31:

tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32
match ip sport 20 0xff flowid 1:31

A filtragem também pode ser feita também a partir da porta de destino alterando o comando "*sport*" por "*dsport*". A lista com as portas padronizadas é encontrada em [IANA, 2009].

4.3.2.2. FILTRAGEM A PARTIR DO PROTOCOLO

É possível filtrar para uma determinada classe alguns protocolos específicos como o TCP, UDP e ICMP. No exemplo a seguir, filtraremos pacotes IGMP, indicados pelo tipo de protocolo 2, para a classe 1:43:

tc filter add dev wlan0 parent 1:0 protocol ip prio 3 u32
match ip protocol 2 0xff flowid 1:43

A lista com os números de cada protocolo pode ser obtida em qualquer distribuição *linux* com a versão *kernel linux* acima de 2.6.24 em diante no arquivo situado em /etc/protocols.

4.3.2.3. FILTRAGEM A PARTIR DO CAMPO TYPE OF SERVICE

O TC torna possível o direcionamento de fluxos de acordo com seu ToS. No exemplo a seguir, o valor de ToS filtrado é 0x28:

tc filter add dev eth0 parent 1:0 prio 3 protocol ip u32
match ip tos 0x28 0xff flowid 1:3

4.3.2.4. FILTRAGEM DE ACORDO COM O ENDEREÇO DE REDE OU DE UMA SUBREDE ESPECÍFICA

A partir deste filtro é possível direcionar fluxos de dados de um endereço IP ou até mesmo de uma subrede inteira, indicando a máscara da subrede. Usaremos o exemplo descrito no tópico DSMARK, onde filtramos todos os fluxos oriundos da sub-rede 192.168.0.7/24 para a classe 1:7:

tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32/
match ip src 192.168.0.7/24 flowid 1:7

Também é possível inserir filtros a partir do endereço de destino substituindo o parâmetro “src” (*source*) por “dst” (*destination*).

4.3.3. FILTROS “FW”

Com o filtro “fw” é possível marcar pacotes a partir do *ipchains* e *iptables* [IPTABLES, 2009], que são ferramentas do *firewall* do *kernel linux*. O exemplo a seguir serve para filtrar o tráfego que está entrando na interface de rede eth1 proveniente da eth0:

tc filter add dev eth1 protocol ip parent 1:0 prio 1 handle 6 fw flowid 1:1

Note que não há nenhuma referência ao filtro u32 no comando. O próximo passo é atribuir o tipo de tráfego passante por cada classe, o que será feito pelo *iptables*, utilizando a tabela *mangle* e o alvo MARK. O a ferramenta *iptables*, que é o *firewall* mais usado em distribuições GNU/Linux e pode ser concatenado com o TC, possui diversas opções de seleção de tráfego.

iptables -A PREROUTING -t mangle -i eth0 -j MARK --set-mark 6

A opção -t é usada para definição da tabela *mangle*. A opção -A é usada para definir a *chain* do pacote. No caso dos roteadores da rede *mesh* utilizaremos a *chain* FORWARD, que aplicará as regras nos pacotes que serão encaminhados por cada roteador. O alvo MARK atribui uma marca aos pacotes com as características definidas pelo comando dado. Essa marca é interpretada pelo TC que faz a reserva de banda para cada tipo de tráfego. A ferramenta *TC MESH* faz uso do *iptables* devido a limitação do TC em filtrar pacotes baseado em seu tamanho.

Na tabela *mangle*, ao contrário das tabelas *filter* e *nat*, as instruções são lidas até o final, mesmo que a condição especificada case com o pacote que está em tráfego. Assim, as regras mais específicas devem ser definidas por último.

4.4. COLETA DE ESTATÍSTICAS DO TC

A ferramenta TC permite obter estatísticas sobre regras de enfileiramento no linux. Infelizmente os resultados estatísticos não são detalhados pelos autores. A seguir são

comentadas as estatísticas da disciplina de enfileiramento HTB. O exemplo seguinte mostra uma disciplina *classful* HTB com três ramificações *classless*. Basicamente, o TC permite a exibição de quantos pacotes/bytes foram enviados nas respectivas filas além de mostrar os pacotes descartados (*dropped*) ou se a fila necessitou atrasar um pacote (*overlimits*) [HUBERT, 2009].

```
# tc -s -d qdisc show dev eth0
 qdisc pfifo 22: limit 5p
 Sent 0 bytes 0 pkts (dropped 0, overlimits 0)

 qdisc pfifo 21: limit 5p
 Sent 2891500 bytes 5783 pkts (dropped 820, overlimits 0)

 qdisc pfifo 20: limit 5p
 Sent 1760000 bytes 3520 pkts (dropped 3320, overlimits 0)

 qdisc htb 1: r2q 10 default 1 direct_packets_stat 0
 Sent 4651500 bytes 9303 pkts (dropped 4140, overlimits 34251)
```

Nas estatísticas de classe podemos verificar todas as informações das estatísticas das regras de enfileiramento:

```
#tc -s -d class show dev eth0
class htb 1:1 root prio 0 rate 800Kbit ceil 800Kbit burst 2Kb/8 mpu 0b
  cburst 2Kb/8 mpu 0b quantum 10240 level 3
 Sent 5914000 bytes 11828 pkts (dropped 0, overlimits 0)
 rate 70196bps 141pps
 lended: 6872 borrowed: 0 giants: 0

class htb 1:2 parent 1:1 prio 0 rate 320Kbit ceil 4000Kbit burst 2Kb/8 mpu 0b
  cburst 2Kb/8 mpu 0b quantum 4096 level 2
 Sent 5914000 bytes 11828 pkts (dropped 0, overlimits 0)
 rate 70196bps 141pps
 lended: 1017 borrowed: 6872 giants: 0

class htb 1:10 parent 1:2 leaf 20: prio 1 rate 224Kbit ceil 800Kbit burst 2Kb/8 mpu 0b
  cburst 2Kb/8 mpu 0b quantum 2867 level 0
 Sent 2269000 bytes 4538 pkts (dropped 4400, overlimits 36358)
 rate 14635bps 29pps
 lended: 2939 borrowed: 1599 giants: 0
```

Overlimits mostra quantas vezes a classe solicitou enviar pacote, mas eles não puderam ser enviados por restrições de taxa/limite superior (*rate/ceil*) (atualmente computada apenas nas classes folha). *Rate (bps* e *pps)* mostra a taxa de bits e pacotes atual (média 10 seg) da classe. *Lended* é o número de pacotes doados por esta classe (de sua taxa) e *borrowed* são pacotes que foram tomados emprestados da classe pai. *Lends* (empréstimos concedidos) são sempre computados na classe que emprestou enquanto *borrows* (empréstimos tomados) são transitivos (quando 1:10 toma emprestado de 1:2 que por sua vez toma emprestado de 1:1 ambos os contadores *borrows* de 1:10 e de 1:2 são incrementados). *Giants* (gigantes) é o número de pacotes maiores que a MTU configurada no comando TC. HTB irá trabalhar com estes pacotes, mas as taxas não serão exatas em todos [HUBERT, 2009].

As demais disciplinas de enfileiramento oferecem a possibilidade da coleta de estatísticas, porém, de forma limitada. Permitindo apenas estatísticas de vazão, pacotes descartados e enviados.

5. FERRAMENTA *TC MESH*

As redes IEEE 802.11 com topologia *mesh* podem oferecer uma grande variedade de serviços, inclusive serviços multimídia, como *streaming* de vídeo e voz sobre IP. Tais serviços necessitam de tratamento diferenciado no que diz respeito à largura de banda, atrasos e perdas de pacotes. A ferramenta *TC MESH* facilita a configuração de parâmetros de QoS através da geração automática de scripts com a sintaxe do controle de tráfego do GNU/*Linux* de acordo com as configurações desejadas pelo usuário. Além disso, os procedimentos de instalação e execução dos scripts nos roteadores são automatizados. Adicionalmente, existe também a possibilidade de visualização gráfica das estatísticas geradas pelo TC em cada roteador. Com essas funcionalidades, o tempo gasto pelo administrador da rede *mesh* para desenvolvimento, instalação, execução dos scripts, início dos testes e análise dos mesmos é reduzido significativamente, permitindo maior agilidade no desenvolvimento e análise da provisão de QoS em redes em malha sem fio.

5.1. FUNCIONAMENTO DO *TC MESH*

A ferramenta *TC MESH* oferece uma interface gráfica web onde o usuário configura os parâmetros desejados para controle do tráfego em cada roteador sem fio da rede em malha ou em um conjunto deles. A interface está integrada com um programa desenvolvido em PHP/XML/HTML que atua automatizando o processo de desenvolvimento das políticas de QoS, integrando em um script todos os parâmetros especificados pelo usuário. Todo o processo pode ser realizado de forma remota, necessitando apenas de autorização dos responsáveis pela rede em malha sem fio em questão para atualização dos roteadores com o script gerado pela ferramenta. O código-fonte da ferramenta se encontra no Anexo 8.1.

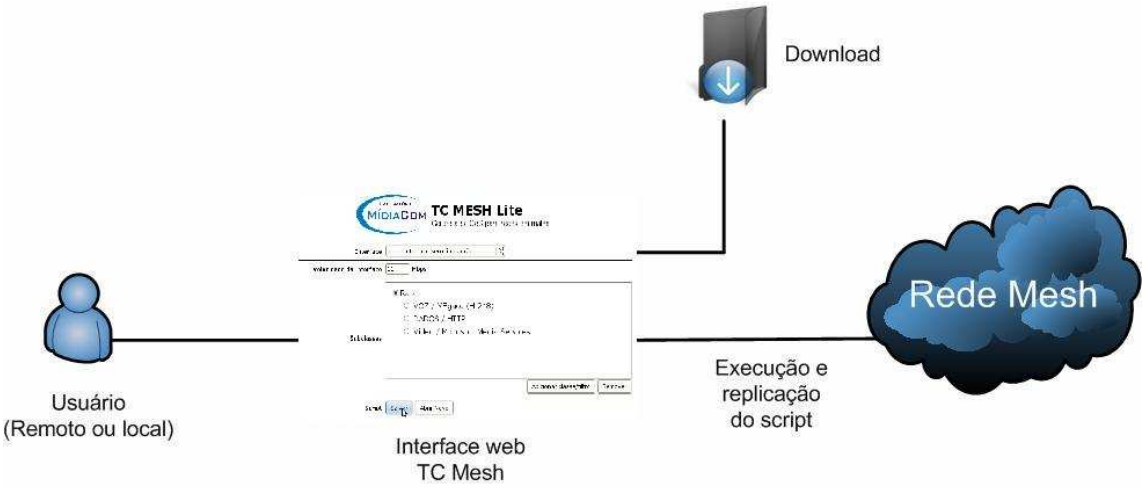

Figura 26. Funcionamento básico do TC MESH.

Como apresentado na Figura 26, é necessário que o usuário da ferramenta configure os parâmetros de QoS desejados através da interface web oferecida pelo *TC MESH*. A partir da geração do script completo pela ferramenta, cabe ao usuário definir qual a finalidade das políticas criadas, seja ela para uso pessoal (*desktop*/*laptops*) através do *download* do script para o computador do usuário, ou para instalação e execução em um ou todos os roteadores da rede em malha sem fio.

A Figura 27 ilustra o funcionamento detalhado da ferramenta *TC MESH*. O usuário remoto necessita de autenticação para uso da ferramenta já que a mesma permite realizar configurações dentro do sistema operacional de cada roteador da rede em malha, e por isso deve ter acesso restrito (passo 1 na Figura 27). Realizada a autenticação, o usuário acessa a ferramenta e configura os parâmetros de QoS de acordo com as suas necessidades. Ao finalizar a configuração, a ferramenta *TC MESH* gera um script TC onde o usuário pode optar pelo *download* para uso posterior nos roteadores ou em um PC, ou pela instalação e execução do mesmo na rede em malha sem fio. Se desejar configurar os roteadores, o script é transferido para o roteador que tem a função de *gateway* da rede em malha sem fio (passo 2 na Figura 27). O *gateway* da rede *mesh* tem conexão cabeada, ligando-o até o servidor de gerência onde se encontra instalada a ferramenta, e uma conexão sem fio, integrando-o à rede em malha. Depois que o script é transferido para o nó *gateway*, é realizada a sua instalação nos outros roteadores da rede em malha sem fio.

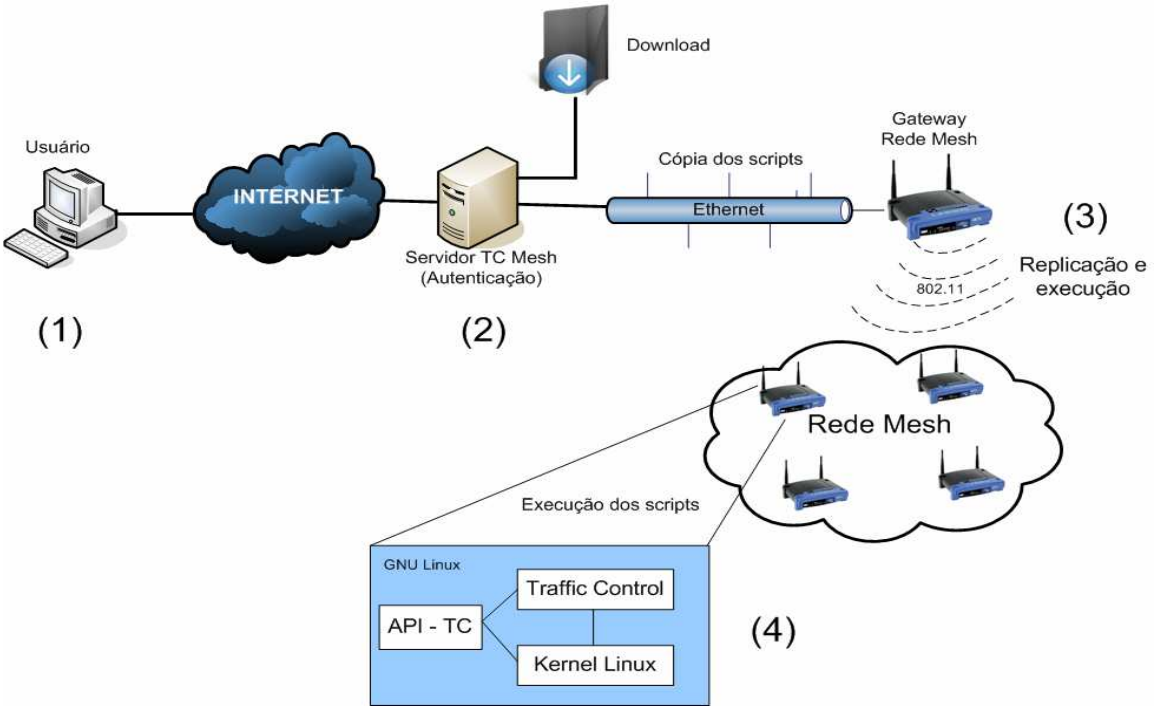

Figura 27. Arquitetura completa do TC MESH.

Caso o usuário deseje copiar para todos os roteadores, a ferramenta *bcp (Broadcast CoPy)* [DUARTE et al, 2007] é utilizada para executar esta função, o *bcp* estabelece várias seções *ssh (Secure Shell)*, uma para cada roteador, enviando a partir destas os scripts para todos os roteadores da rede *mesh* (passo 3 na Figura 27). Caso o usuário queira configurar um roteador específico, usa-se o software *scp* (Secure CoPy), disponível na maioria das distribuições GNU/Linux. Com os scripts disponíveis em cada roteador, a ferramenta *bshell (broadcast Shell)* [DUARTE et al, 2007] realiza a execução dos scripts em todos os roteadores. O *bshell* executa em cada roteador o comando necessário para a execução do script criado pelo usuário. Basicamente, cada script será interpretado pelo software TC dos roteadores e as informações serão repassadas para a API (*Application Programming Interface)* do software, que se encarregará de passar as instruções para o *kernel linux* dos roteadores e aplicar as políticas de QoS em cada roteador (passo 4 na Figura 27).

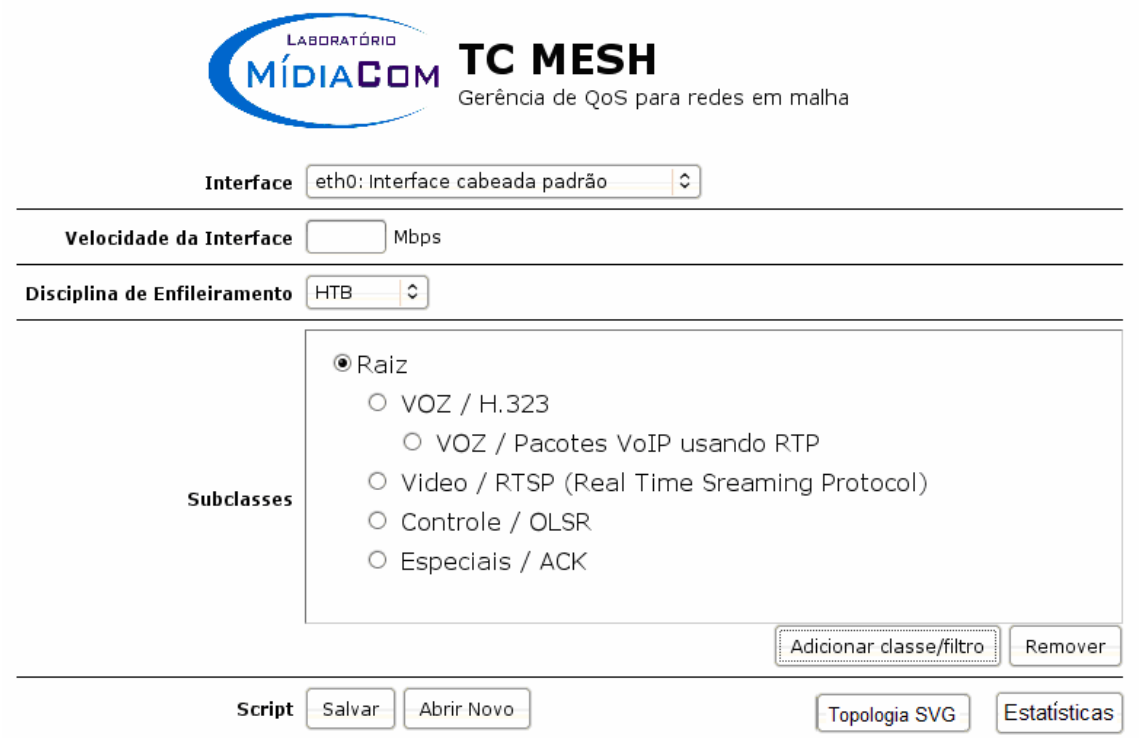

Figura 28. Tela principal do TC MESH.

A Figura 28 apresenta a tela principal da ferramenta *TC MESH*. Para o usuário desenvolver as políticas de QoS desejadas, são necessárias seis etapas: definição da interface de rede, taxa máxima da interface, estabelecimento de disciplina de enfileiramento, criação de classes/subclasses e estabelecimento de filtros para cada uma delas. A definição da interface de rede é necessária para identificar para a ferramenta em qual dispositivo de rede do roteador *mesh* ou outra interface de rede de algum outro equipamento serão aplicadas as políticas, definindo assim a vazão máxima daquela interface de acordo com a tecnologia de rede aplicada (ex: 11 Mbps para redes sem fio IEEE 802.11b). O estabelecimento de uma disciplina de enfileiramento, como o nome já diz, identifica qual será a disciplina de enfileiramento principal, sendo a partir dela possível a criação de classes e subclasses, dependendo da disciplina que será usada. *TC MESH* usa como disciplinas de enfileiramento FIFO (*First In First Out*), SFQ (*Stocastic Fair Queueing*), TBF (*Token Bucket Filter*), RED (*Randon Early Detecction*), PRIO (*Priority Queue*), HTB (*Hierarquical Token Bucket*) e DSMARK (*Diffserv Mark*), descritas no Capítulo 4.

Para cada classe/subclasse criada, é possível a vinculação de filtros através do botão “Adicionar classe/filtro”, definindo-se assim quais fluxos e qual a prioridade daquele fluxo

dentro de uma determinada classe. A ferramenta limita a inclusão em dezesseis (16) classes e, em cada classe, a criação de mais oito (8). Esta limitação serve para não sobrecarregar os processadores dos roeadores e, consequentemente, não reduzir a eficiência das classes criadas.

A ferramenta *TC MESH* opera em dois modos para criação de classes e configuração de filtros: básico e avançado. O modo básico foi desenvolvido para usuário com poucos conhecimentos da sintaxe do TC e é baseado nos serviços que o usuário deseja classificar. Já o modo avançado oferece uma grande variedade de combinações de filtros permitindo concatenar classificações a partir da porta, endereço IP, protocolo utilizado, campos *ToS* e *DS Field* e tamanho do pacote. É possível visualizar em tempo real quais classes/subclasses estão sendo criadas através da caixa "subclasses" (Figura 28), organizando de forma hierárquica cada classe criada. Na página inicial da ferramenta existe também a possibilidade do usuário visualizar a topologia da rede em malha que está sendo gerenciada e a qualidade de cada enlace através do botão "Topologia SVG", onde será encaminhado para a ferramenta *Mesh Topology Viewer* (MTV) [VALLE et al, 2008]. Com o botão "Estatísticas" o usuário é encaminhado para o local onde os gráficos com os parâmetros do TC de cada roteador estão localizados. É possível também salvar ou abrir um script criado anteriormente com os botões "salvar" e "abrir novo", respectivamente.

Realizados os seis passos descritos anteriormente, a ferramenta *TC MESH* gera um script TC com os parâmetros especificados pelo usuário juntamente com um *log* das ações realizadas pelo usuário, como exibido na Figura 29. Esse *log* permite ao usuário um controle maior das ações que ele executou para desenvolver aquela política de filas. Além disso, todos os scripts gerados pelo *TC MESH* têm comentários em todas as ações realizadas pelo script, demonstrando aos usuários exatamente o que cada linha do script executa nos dispositivos de rede, servindo também para o aprendizado no desenvolvimento manual de scripts. Há ainda a possibilidade de adaptar o script para o uso em computadores pessoais, clicando no botão "Versão para PC". Esta opção existe devido a uma diferença entre o *kernel linux* do roteador (mais antigo) e os dos computadores (normalmente mais novos), e com isso é necessária a inclusão na versão para roteadores de linhas de código adicionais para habilitar os módulos das disciplinas de enfileiramento. Como último passo, basta o usuário escolher entre salvar em disco, instalar e executar em todos os roteadores da rede em malha ou escolher algum roteador específico a ser aplicado aquele script. É importante ressaltar que para o envio do script para todos os roteadores, é necessário que os mesmos tenham as mesmas interfaces de rede, como no caso da rede em malha usada para os testes desta dissertação.

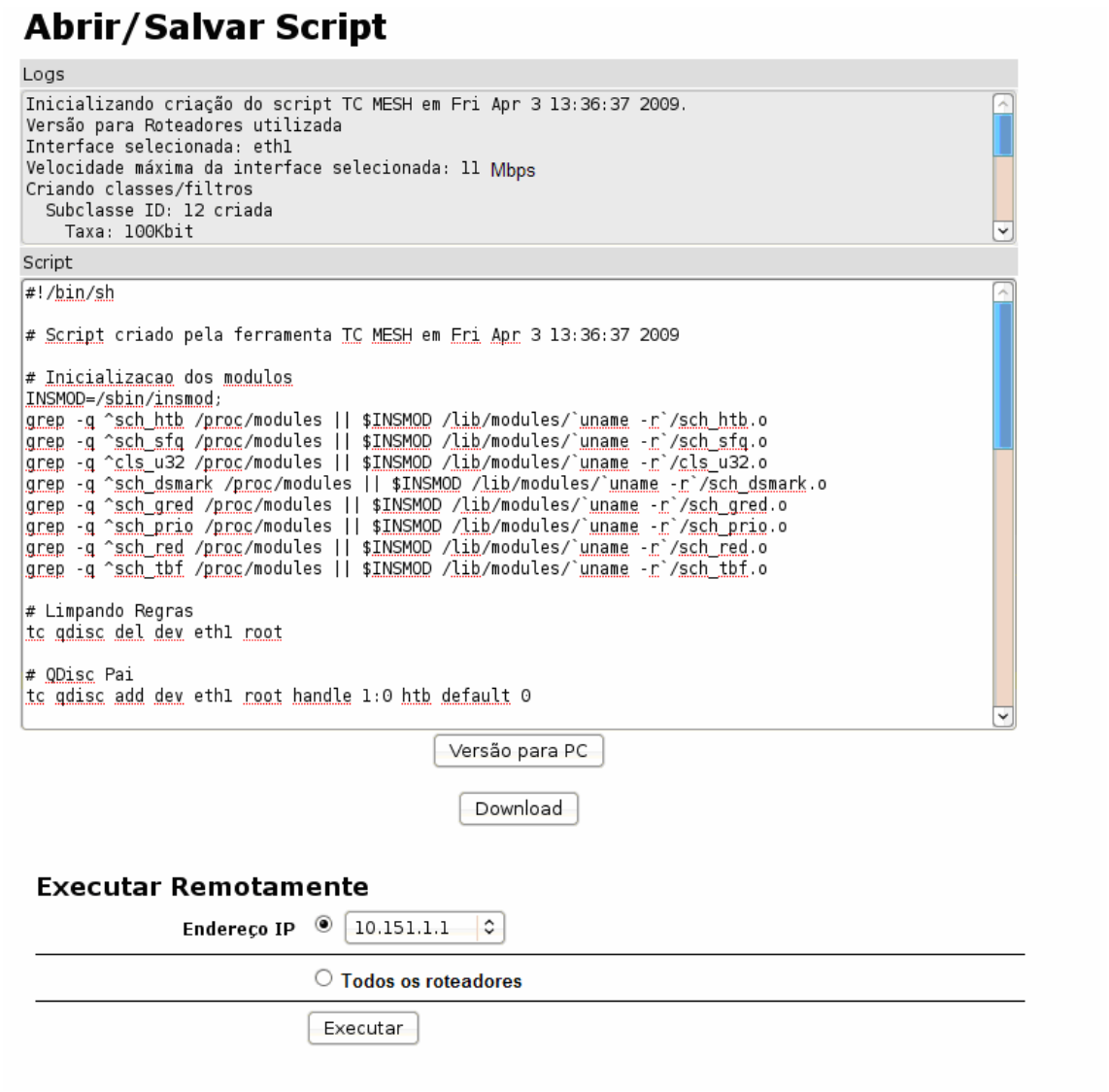

Figura 29. Script TC gerado pela ferramenta.

Uma vantagem do *TC MESH* com relação às outras ferramentas de QoS [CARVALHO e ABELÉM, 2008] [TCNG, 2008] é a sua interface mais intuitiva que as oferecidas pelas demais devido a ferramenta exibir ao usuário todas as etapas do desenvolvimento do script além de explicar ao usuário detalhadamente o que o script irá configurar no dispositivo de rede. Outra vantagem é a automatização do processo de instalação e execução remota das configurações diretamente nos roteadores da rede em malha, o que reduz significativamente o tempo necessário para reconfiguração das políticas nos roteadores da rede sem fio, não necessitando da interação direta dos administradores com cada roteador. Além disso, a ferramenta limita o acesso dos usuários aos roteadores para

realizar ações de QoS, o que aumenta a segurança da rede, restringindo o número de pessoas autorizadas que acessam os mesmos.

Apesar de *TC MESH* ter sido desenvolvida e estar sendo utilizada em redes em malha sem fio, ela pode ser útil para configuração de parâmetros de QoS em outras redes que utilizam roteadores IP com sistema operacional baseado em GNU/*Linux* e que permitem a utilização do software TC.

5.1.1. TC MESH MODO BÁSICO

Apesar do *TC MESH* já oferecer uma grande facilidade no manuseio de políticas de QoS, os usuários necessitam de conhecimentos básicos sobre conceitos como disciplinas de enfileiramento e definição de filtros no TC para manipular os campos específicos do datagrama IP. Como alternativa para usuários com pouquíssimo ou nenhum conhecimento sobre o TC, foi desenvolvida na opção “classes/filtros” da ferramenta o modo de operação básico (Figura 30).

Figura 30. Exemplo do modo básico do TC MESH.

Com o modo básico é possível criar com ainda maior facilidade políticas de QoS, pois o usuário não necessita saber nenhum aspecto específico do TC e sim os conceitos genéricos de qualidade de serviço**.** Com isso, apesar de limitar significativamente as funcionalidades da ferramenta, o usuário ganha maior confiabilidade de que os scripts que desenvolvidos atendem as suas necessidades e reduzem a chance de erro na configuração. Neste modo de funcionamento da ferramenta *TC MESH*, o usuário escolhe que tipo de serviço deseja filtrar. *TC MESH* oferece cinco classificações: voz, vídeo, dados, controle e especiais. Na opção “voz”, por exemplo, é possível utilizar a opção “Sub-Filtro” para classificar pacotes da suíte

H.323 e SIP além de alguns softwares proprietários como Skype e MSN. Em “vídeo” o usuário pode especificar streamings de vídeo e Windows Media Services, por exemplo. Já em “dados”, os pacotes oriundos dos protocolos FTP, TFTP e HTTP podem ser direcionados para uma classe específica. Em “controle” vários protocolos de controle podem ser classificados como os pacotes de sinalização do OLSR (Optimized Link State Routing), AODV (Ad Hoc On Demand Distance Vector) e NAT (Network Address Translator). Os filtros “especias” tratam-se de serviços como AOL Messenger, Yahoo Messenger, pacotes marcados com ACK e qualquer outro que não possa ser catalogado nas opções anteriores. Vale lembrar que os serviços oferecidos pelo *TC MESH* foram catalogados pela Internet Assigned Numbers Authority (IANA) [IANA, 2009], utilizando a concatenação de filtros por porta e tipo do protocolo.

Uma facilidade importante do *TC MESH* é a possibilidade de inclusão de novos filtros através de um arquivo de configuração XML, como ilustrado na Figura 31, bastando que o usuário tenha conhecimentos sobre a sintaxe do TC para adicionar o filtro desejado e nomeá-lo. O Anexo 8.2 apresenta o arquivo XML atualmente usado pela ferramenta. Nesse arquivo de configuração da ferramenta *TC MESH* em XML, um usuário mais experiente no uso do TC pode adicionar a sintaxe de um novo filtro para expandir a quantidade de filtros pré-estabelecidos pela ferramenta *TC MESH* no modo básico. O conteúdo do arquivo XML é convertido em *arrays* através do aplicativo MiniXML [MiniXML, 2009] para permitir o uso dos filtros nos scripts.

```
- <tc>
  - <filters>
    - <filter>
        <name>VOZ</name>
      - <subfilter>
          <name>Megaco (H.248)</name>
          <command>tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\ match ip protocol 17 0xff\ match ip
            sport 2945 0xff \ match ip sport 2944 0xff \ flowid 1:{id}</command>
        </subfilter>
      - <subfilter>
          <name>H.323</name>
          <command>tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\ match ip protocol 17 0xff\ match ip
            protocol 06 0xff\ match ip sport 1300 0xff \ flowid 1:{id}</command>
        </subfilter>
      - <subfilter>
          <name>SIP</name>
          <command>tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\ match ip protocol 17 0xff\ match ip
            protocol 06 0xff\ match ip sport 5060 0xff \ flowid 1:{id}</command>
        </subfilter>
      - <subfilter>
          <name>Pacotes VoIP usando RTP</name>
          <command>tc filter add dev {interface} parent 1:0 protocol ip prio 0 handle 9 fw flowid 1:{id} iptables -t mangle -A
            FORWARD -p udp -m length --length 40:160 -j MARK --set-mark 0x9</command>
        </subfilter>
      </filter>
```

Figura 31. Exemplo do arquivo XML.

5.1.2. TC MESH MODO AVANÇADO

O modo avançado da ferramenta *TC MESH* oferece aos usuários mais familiarizados com os conceitos de controle de tráfego do GNU/Linux maiores recursos para desenvolver os seus próprios filtros de acordo com as opções possíveis da ferramenta. Neste modo, o usuário pode desenvolver o filtro para uma determinada classe de acordo com os parâmetros dos cabeçalhos IP/TCP/UDP ou até mesmo pelo tamanho do pacote. A importância de oferecer ao usuário essa flexibilidade se dá pela necessidade de classificar pacotes específicos ou até mesmo criados pelos próprios usuários. Com a abordagem avançada é permitido concatenar vários parâmetros como porta e tipo de protocolo, *Type Of Service*, endereço IP de origem e destino, ou até mesmo um tamanho específico de pacote como em aplicações voz sobre IP (VoIP) onde, dependendo do *codec* usado, existe uma variação do tamanho do pacote.

As Figuras 32 e 33 ilustram a interface avançada do *TC MESH*, permitindo escolher se o usuário deseja usar os cabeçalhos TCP/IP e/ou tamanho do pacote através da opção “Tipo de Filtro”. Nesta interface o usuário pode manipular os filtros de acordo com o endereço IP origem/destino, porta origem/destino, campo *Type Of Service*, protocolos (especificados no diretório /etc/protocols de qualquer distribuição Linux), tamanho mínimo e máximo do pacote

em *bytes*, além das taxas garantida, máxima e a possibilidade de compartilhar (“Emprestar?”) a banda que não está sendo utilizada (*borrowing)*, no caso do mecanismo de escalonamento HTB. Vale lembrar que é possível concatenar os filtros baseados no cabeçalho TCP/IP juntamente com os baseados no tamanho do pacote.

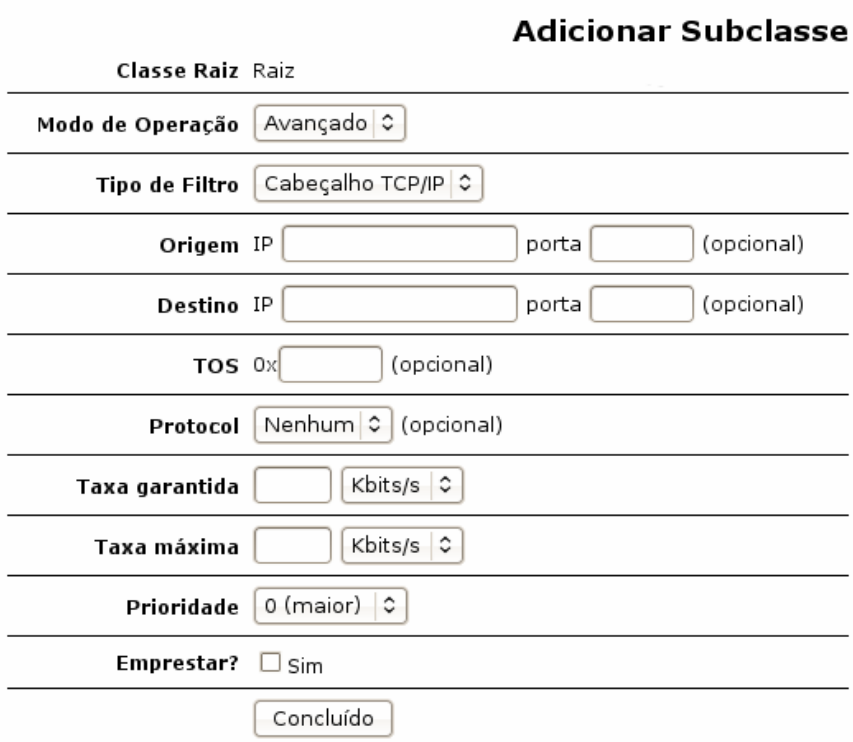

Figura 32. Exemplo do modo de operação avançado do TC MESH.

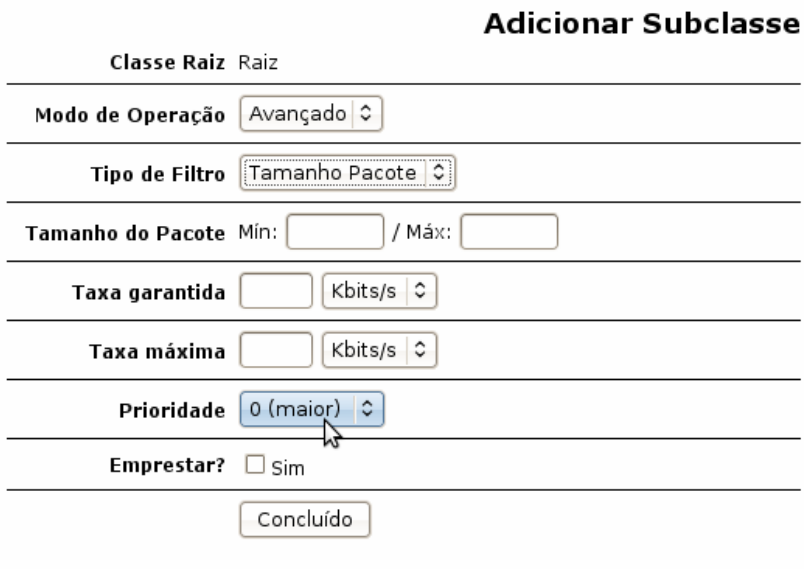

Figura 33. Exemplo 2 do modo de operação avançado do TC MESH.

5.2. ESTRUTURA INTERNA DA FERRAMENTA *TC MESH*

A estrutura interna da ferramenta *TC MESH* está ilustrada na Figura 34.

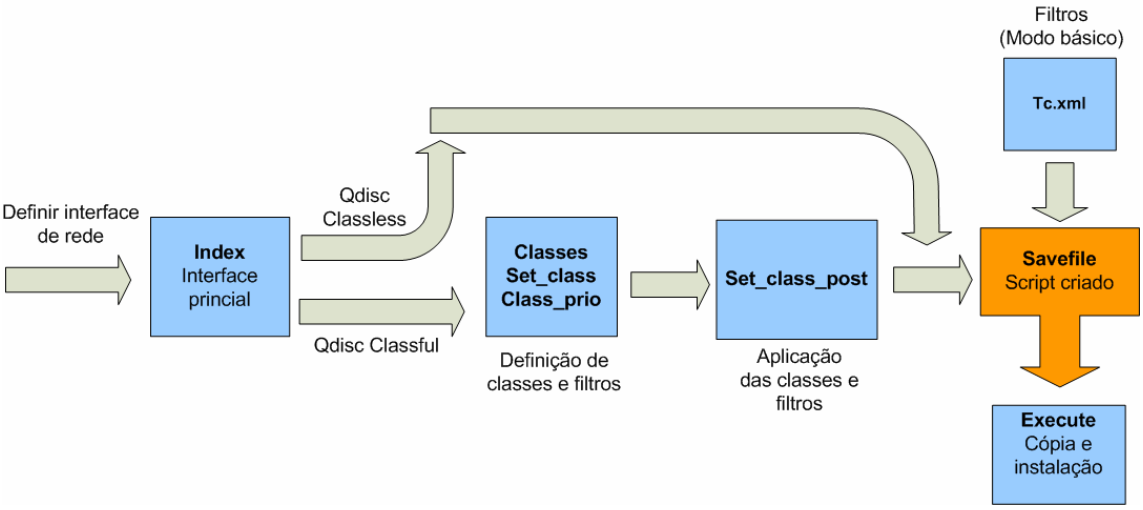

Figura 34. Estrutura interna do TC MESH.

Inicialmente, o usuário acessará a interface principal da ferramenta implementada pelo arquivo “index.php” para definir a interface de rede a ser usada juntamente com qual disciplina de enfileiramento deseja-se incluir no script. Caso o usuário opte por uma disciplina de enfileiramento *classful*, serão oferecidas as opções para definição de classes juntamente com os filtros (módulos “classes.php”, “set_class.php” e “class_prio.php”). Existe um caso especial na definição de classes e filtros envolvendo a disciplina de enfileiramentro PRIO, pois como a mesma tem uma número fixo de classes foi criada um módulo específico para tal, tratando-se do “class_prio.php”. Este módulo evita que o usuário possa criar classes adicionais, não permitindo que haja scripts desenvolvidos de forma errada. Caso uma *qdisc classless* seja usada, o script não permite a atribuição de classes e filtros e automaticamente cria o script para o usuário através do módulo “savefile.php”. Assim que o usuário especifica todos os parâmetros necessários para a definição das classes e filtros o módulo “set_class_post.php” agrega essas informações no futuro script do usuário de acordo com a sintaxe do TC. Caso o usuário esteja usando o modo de configuração básico da ferramenta, o arquivo XML “tc.xml” é acessado para obter os filtros definidos pelo usuário para finalizar a criação do script para que finalmente o script completo seja criado pelo módulo “savefile.php”. Finalizado o script, o usuário pode optar por instalá-lo em algum roteador

através do *Broadcast Copy* e *Broadcast Shell* ou salvar localmente no computador no qual o usuário esteja acessando a ferramenta a partir do módulo “execute.php”.

5.3. COLETA E VISUALIZAÇÃO DE ESTATÍSTICAS

O *TC MESH* também permite a visualização de forma gráfica das estatísticas de cada classe especificada nos roteadores. Basicamente, a ferramenta oferece dados de vazão em cada classe, pacotes perdidos e quantidade de bytes enviados e *overlimits*, que se trata dos pacotes que excederam a sua cota. A possibilidade de visualizar de forma gráfica esses parâmetros permite uma maior clareza para a compreensão do comportamento de determinada disciplina de enfileiramento e classes em uso em uma rede IP. Além disso, a facilidade de coleta automática dos dados estatísticos em tempo real e a geração de gráficos facilita a análise das políticas de QoS em uso e agilizando uma eventual mudança necessária, caso o script não atenda as necessidades dos usuários.

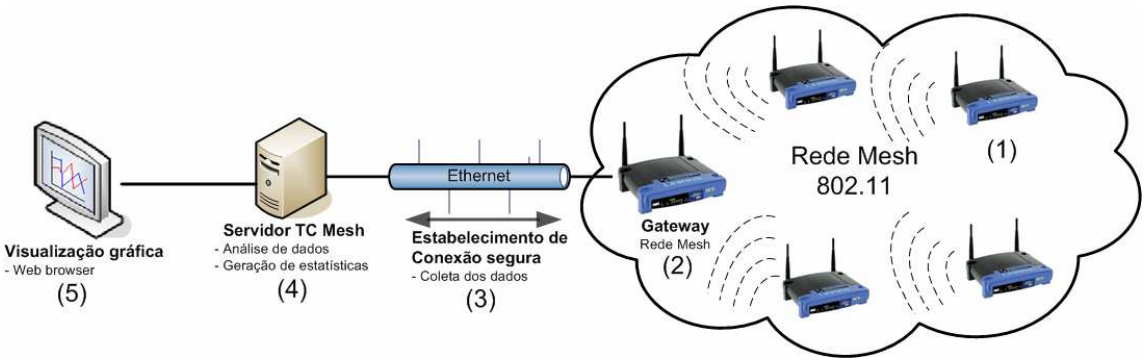

Figura 35. Procedimento para coleta, análise e visualização de estatísticas.

Como descrito na Figura 35, para a visualização das estatísticas primeiramente é necessário coletar as informações geradas pelo TC em cada um dos roteadores (passo 1). Os dados coletados em cada roteador sobre as informações de QoS são armazenados em um arquivo texto e enviados até o *gateway* da rede para a posterior coleta destes dados pela ferramenta *TC MESH* (passo 2). Após isso, o servidor *TC MESH* estabelece conexão em intervalos entre dez e quinze minutos com o *gateway* para coletar os dados (passo 3). Este intervalo de tempo foi estabelecido como sendo o mais adequado para não sobrecarregar o processamento do roteador no envio dos arquivos. Assim que os arquivos de texto chegam ao servidor, são tratados pelo *TC MESH* para serem extraídos somente os dados relevantes para a geração dos gráficos. O software Matplotlib [MATPLOTLIB, 2009], é usado para a

interpretação destes arquivos e para a geração dos gráficos (passo 4). Por fim, o usuário poderá visualizar os gráficos gerados de todos os roteadores da rede sem fio em questão através de um *web browser* (passo 5). A ferramenta *TC MESH* gera gráficos de todos os roteadores em períodos horários (última hora), diários, semanais, mensais e anuais. Com esses dados, o usuário da ferramenta pode analisar o tráfego da rede com agilidade.

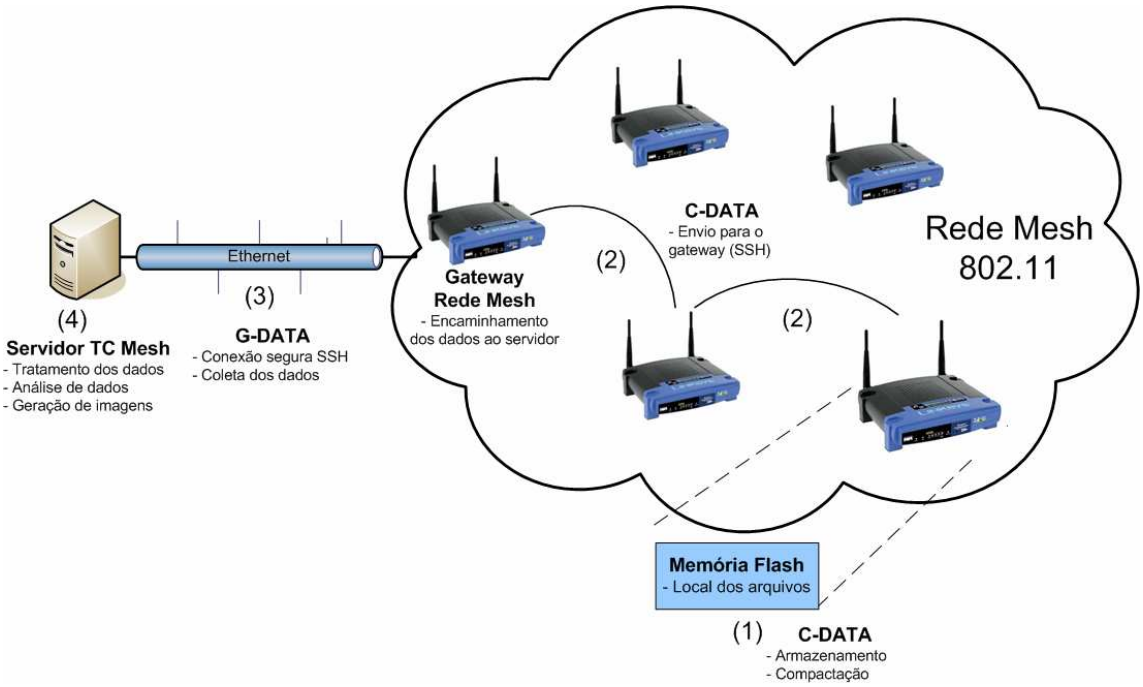

Figura 36. Procedimento para coleta de estatísticas na rede.

O processo de coleta das estatísticas é exibido com maiores detalhes na Figura 36. Em cada roteador da rede em malha sem fio foi desenvolvido um script intitulado *Copy DaTa* (C-DaTa) com a função de capturar e armazenar as informações estatísticas que o TC oferece em relação a disciplinas de enfileiramento e classes. O C-DATA armazena esses dados na memória *flash* do roteador em arquivos texto em intervalos de quinze segundos, copiando a saída do comando do TC necessária para a visualização das estatísticas mostrados no *prompt* de linha de comando e comprimindo os diversos arquivos de texto em um único arquivo com o formato “tar.gz” a cada dez minutos, facilitando o envio posterior dos dados para o *gateway* (passos 1 e 2). Para o próximo passo, foi desenvolvido um script intitulado Get *Data* (G-DATA), onde o servidor *TC MESH* acessa o *gateway* em períodos de dez a quinze minutos para coletar as informações estatísticas de todos os roteadores através de uma sessão SSH (*Secure SHell*), armazenar no servidor e descompactar os arquivos (passo 3). Caso ocorra

congestionamento na rede que impeça ou dificulte a transmissão das informações estatísticas na rede em malha, todos os scripts desenvolvidos com o uso da disciplina de enfileiramento HTB para o uso nos roteadores contêm uma classe específica com nível de prioridade máxima para pacotes SSH, que usam porta 22, tanto UDP quanto TCP conforme [IANA, 2009]. Segundo testes realizados na rede em malha da UFF, o tráfego dos dados estatísticos da rede não ultrapassa 1% da capacidade de um enlace de 1 Mbps, sendo que cada arquivo tem uma tamanho médio de 5 KB. Após a chegada dos dados no servidor, o *TC MESH* trata os arquivos textuais, removendo as informações desnecessárias e formatando o arquivo texto através da ferramenta AWK [AWK, 2009], que fornece a função de selecionar os dados a serem preservados dentro do arquivo de acordo com certos padrões especificados no programa. Neste caso, todos os valores numéricos precedidos das palavras *rate, packets, drop e overlimits* são aproveitados e organizados no arquivo de forma que o Matplotlib compreenda. Basicamente, os valores a serem plotados necessitam estar alinhados em uma coluna para que possam ser sequencialmente identificados pelo Matplotlib, sendo que cada valor corresponde a um ponto no gráfico. Assim, o Matplotlib cria imagens de acordo com os valores passados pela ferramenta, atualizando-as assim que mais dados chegam de cada roteador (passo 4). Na figura 37 temos a interface principal para a seleção dos gráficos das estatísticas da rede. Já nas Figuras 38 e 39, temos exemplos de gráficos de vazão horário e semanal, respectivamente. Os gráficos são organizados de acordo com o número de classes/subclasses, oferecendo um gráfico para cada parâmetro de acordo com sua unidade (Mbps, bytes, overlimits, etc). Com esta funcionalidade é possível realizar uma análise mais detalhada do comportamento das disciplinas de enfileiramento e classes na rede.

Figura 37. Interface de seleção de gráficos.

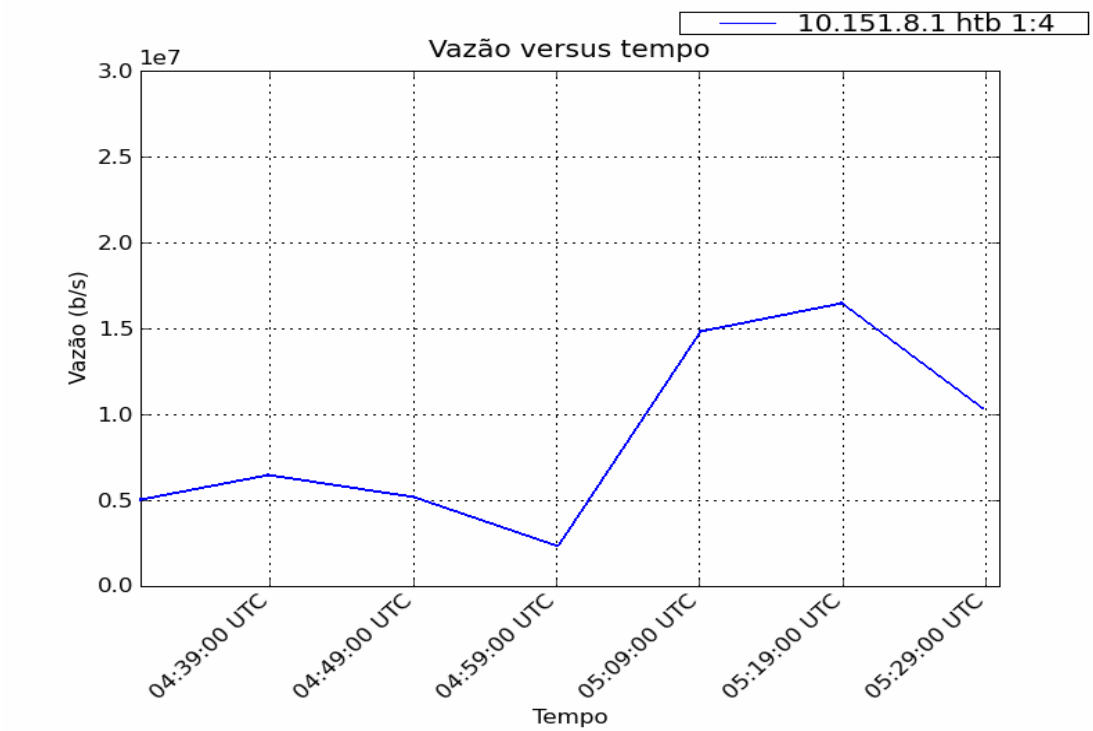

Figura 38. Vazão em tempo real da disciplina de enfileiramento raiz do gateway da rede mesh (horário).

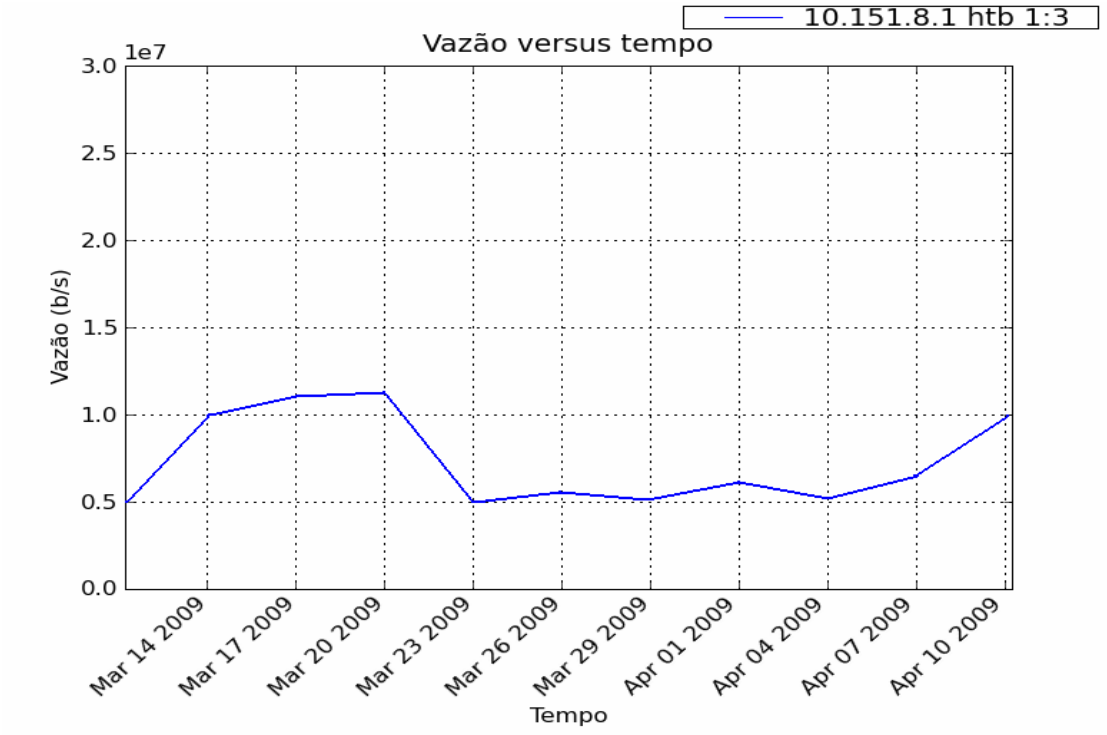

Figura 39. Vazão em tempo real da disciplina de enfileiramento raiz do gateway da rede mesh (semanal).

O diferencial desta proposta em relação ao tradicional protocolo de gerência de redes Simple Network Management Protocol (SNMP) [RFC 1157, 2009] é a redução da carga de processamento em cada roteador, centralizando todo o processo de tratamento dos dados estatísticos dos roteadores no servidor, que contém muito mais poder de processamento e memória. No SNMP, o agente (presente em cada roteador) necessita administrar uma *Management Information Base* (MIB), que consiste em um banco de dados que contém o conjunto de parâmetros que procuram abranger todas as informações pertinentes ao roteador. Tais informações são centralizadas em cada roteador, que necessita de espaço em disco para primeiramente instalar no cliente SNMP e em seguida ter processamento suficiente para executar esse cliente além de desempenhar as outras funções (transmissão/recepção, execução de outros programas, por exemplo). No caso dos roteadores Linksys WRT54GL usados na rede em malha sem fio da UFF, o espaço em disco disponível para instalação do SMNP é bastante limitado já que o mesmo tem apenas 4 MB para instalação de todos os itens incluindo o sistema operacional. Em relação ao processamento, o roteador tem apenas um processador com clock de 200Mhz. Em contraste, o servidor *TC MESH*, possui um

processador de última geração com quatro núcleos trabalhando com um clock de 2,33GHz, 4GB de memória RAM e 500GB de espaço em disco.

A proposta do *TC MESH* também é mais robusta que outra ferramenta muito usada pela comunidade acadêmica, o RRDtool [RRDTOOL, 2009], pois esta necessita de uma sincronização precisa entre os relógios do servidor com os roteadores. Não havendo tal sincronização, poderá ocorrer perdas ou imprecisão das informações fornecidas. No caso de uma rede em malha sem fio, esse sincronismo torna-se difícil principalmente para os nós que ficam a muitos saltos de distância do *gateway*, acarretando um atraso significativo na chegada dos dados ao servidor em comparação com os roteadores mais próximos ao *gateway*. Mesmo com os relógios sincronizados, existe o risco dos roteadores reiniciarem e o relógio ser reiniciado, prejudicando a confiabilidade do gerador de estatísticas. Além disso, não é possível a instalação de algum tipo de *software* para sincronização dos relógios entre os roteadores devido às limitações de memória. Com a proposta de geração de estatísticas do *TC MESH*, o horário é ajustado de acordo com o relógio do servidor, desconsiderando o atraso de alguns segundos na chegada dos dados. Assim, mesmo após ocorrer algum tipo de pane em um roteador, no momento que seu funcionamento for normalizado ele continua a fornecer os dados para o *TC MESH* sem a necessidade de nenhum tipo de sincronização.

5.4. VALIDAÇÃO DA FERRAMENTA

Com o objetivo de averiguar a eficiência da ferramenta *TC MESH* no desenvolvimento de scripts com políticas de QoS, foram desenvolvidos testes demonstrando de que forma a ferramenta em conjunto com o controle de tráfego linux pode ajudar a melhorar a qualidade de serviço para determinados fluxos. Os testes serão descritos nas seções a seguir.

5.4.1. TESTE 1: PRIORIZAÇÃO DE PACOTES DO PROTOCOLO OLSR

Neste primeiro teste, foi desenvolvido um script com o objetivo de oferecer através da disciplina de enfileiramento HTB prioridade máxima aos pacotes do protocolo de roteamento OLSR. Este protocolo utiliza a porta UDP 698, segundo [IANA, 2009]. É a partir desta informação que o pacote será classificado para a classe mais prioritária (1:1) e os demais pacotes são encaminhados para uma classe de menor prioridade (1:2). A Figura 40 ilustra a estrutura de classes deste teste, onde foi oferecido ao OLSR uma vazão mínima de 500 Kbps, entretanto, conforme o mecanismo de *borrowing* do HTB, caso o OLSR necessite de mais banda ele poderá requisitar à classe vizinha devido a sua maior prioridade.

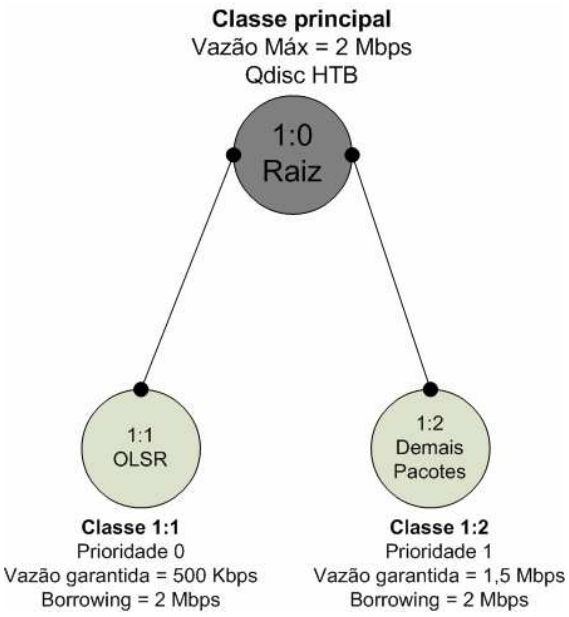

Figura 40. Estrutura do script para priorização do OLSR.

Para a realização dos testes foi usada a rede em malha sem fio do projeto Remesh [REMESH, 2009], onde é possível visualizar a topologia da rede e a qualidade dos enlaces através da ferramenta MTV (Mesh Topology Viewer) [VALLE et al, 2008]. Como ilustrado na Figura 41, MTV oferece uma legenda que indica a qualidade de cada enlace da rede, onde as linhas vermelhas indicam pior qualidade e as linhas azuis escuras o melhor caso possível.

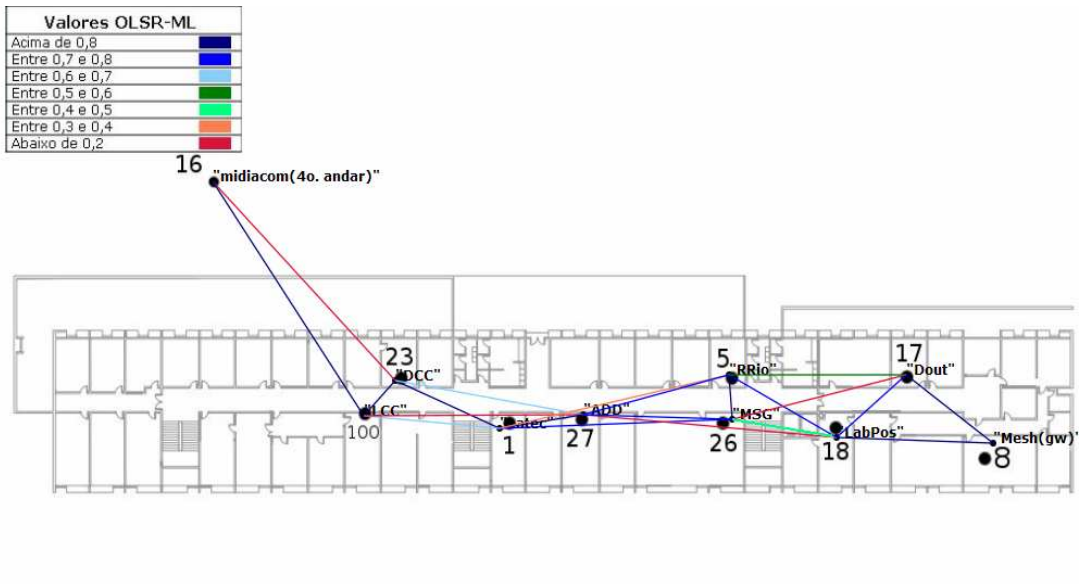

Figura 41. Topologia de rede mesh sem o uso do TC.

No cenário da Figura 41, nenhum tipo de prioridade está sendo oferecida aos pacotes de sinalização OLSR contribuindo para alguns enlaces fiquem com baixo nível de qualidade segundo a métrica *Minimum Loss* como, por exemplo, o enlace entre os nós 26 e 17. Desta forma, a comunicação entre eles poderá ficar comprometida. Na Figura 42 temos a topologia da rede usando um script TC desenvolvido pelo *TC MESH* como mostrado na Figura 40.

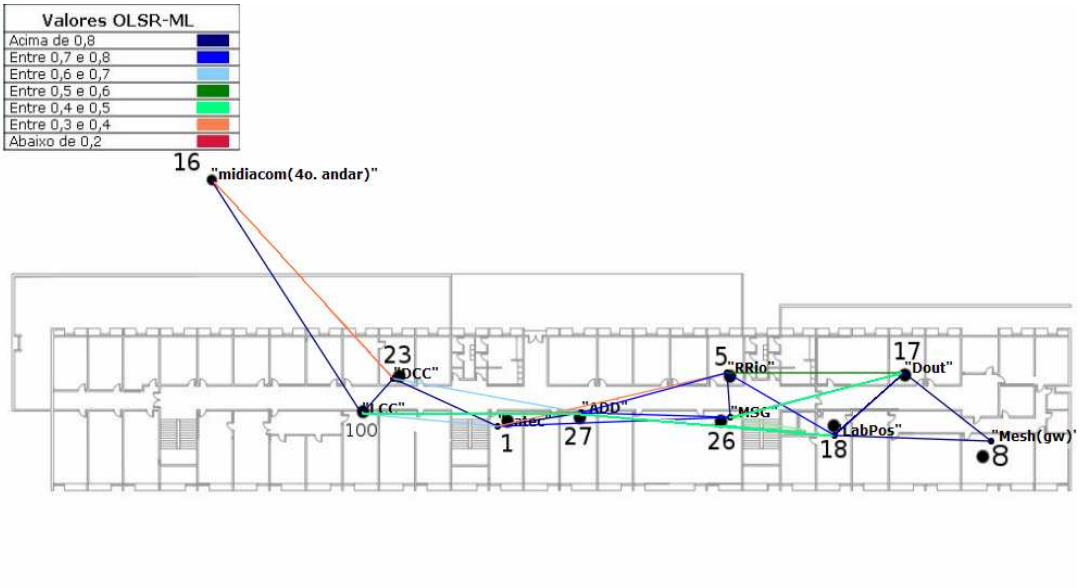

Figura 42. Topologia da rede mesh usando o TC.

Note que, na Figura 42, fica evidente a melhora na qualidade do enlace entre os nós 26 e 17 e de vários outros enlaces, que no novo cenário possuem cor verde e não mais vermelha. Através do script desenvolvido com a ferramenta *TC MESH*, o protocolo OLSR foi classificado com prioridade máxima, por isso foi possível obter melhoras significativas na medida de qualidade dos enlaces.

5.4.2. TESTE 2: TRATAMENTO DIFERENCIADO PARA FLUXOS DISTINTOS

O objetivo do segundo teste foi demonstrar com o auxílio da ferramenta de estatísticas do *TC MESH* o comportamento de determinadas classes e suas respectivas prioridades. Para este teste foi desenvolvido um script através do *TC MESH* usando a disciplina de enfileiramento HTB com três classes. Em cada classe, existe um filtro para um tipo de pacote específico como mostra a Figura 43: pacotes UDP usando a porta 2001 (maior prioridade), pacotes UDP usando a porta 2002 (prioridade intermediária) e pacotes UDP usando a porta

2003 (menor prioridade). Para a geração dos pacotes com tais características e com a vazão desejada foi utilizada a ferramenta *iperf* [IPERF, 2009]. Os testes foram realizados entres os nós 8 e 18 da rede em malha sem fio (Figura 42) com o script instalado em ambos os roteadores. Assim como no teste descrito anteriormente, a vazão na interface de rede sem fio foi limitada a 2 Mbps.

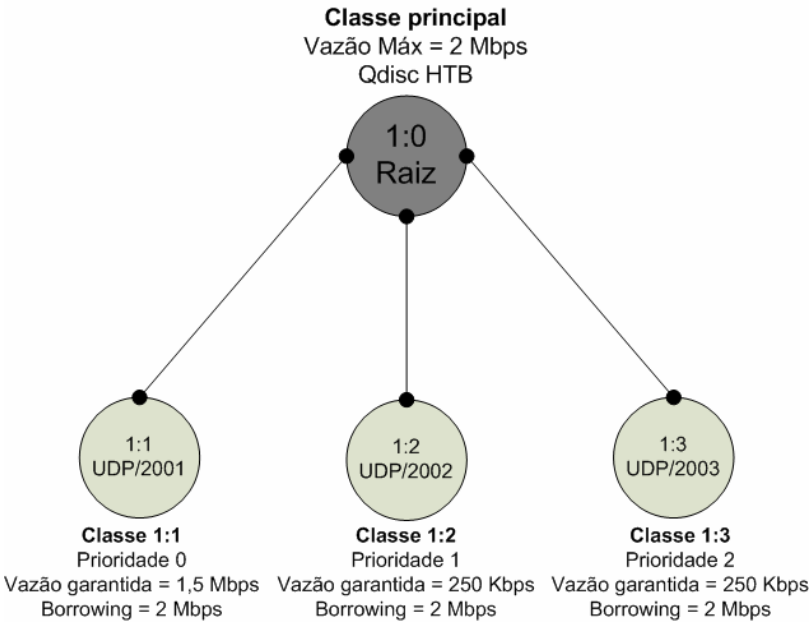

Figura 43. Estrutura do script gerado para o teste 2.

Inicialmente, foram estabelecidas vazões para as classes menos prioritárias acima da taxa garantida para cada uma. Foram estabelecidas as vazões de 1 Mbps e 500 Kbps durante cento e vinte minutos para as classes 1:2 e 1:3, respectivamente. Neste teste, evidencia-se o funcionamento do mecanismo de *borrowing* do HTB, pois há o empréstimo da porcentagem de banda pertencente à classe 1:1, que está no momento subutilizando a sua banda reservada, para as demais classes. A Figura 44 mostra os gráficos de vazão de cada classe dos primeiros sessenta minutos.

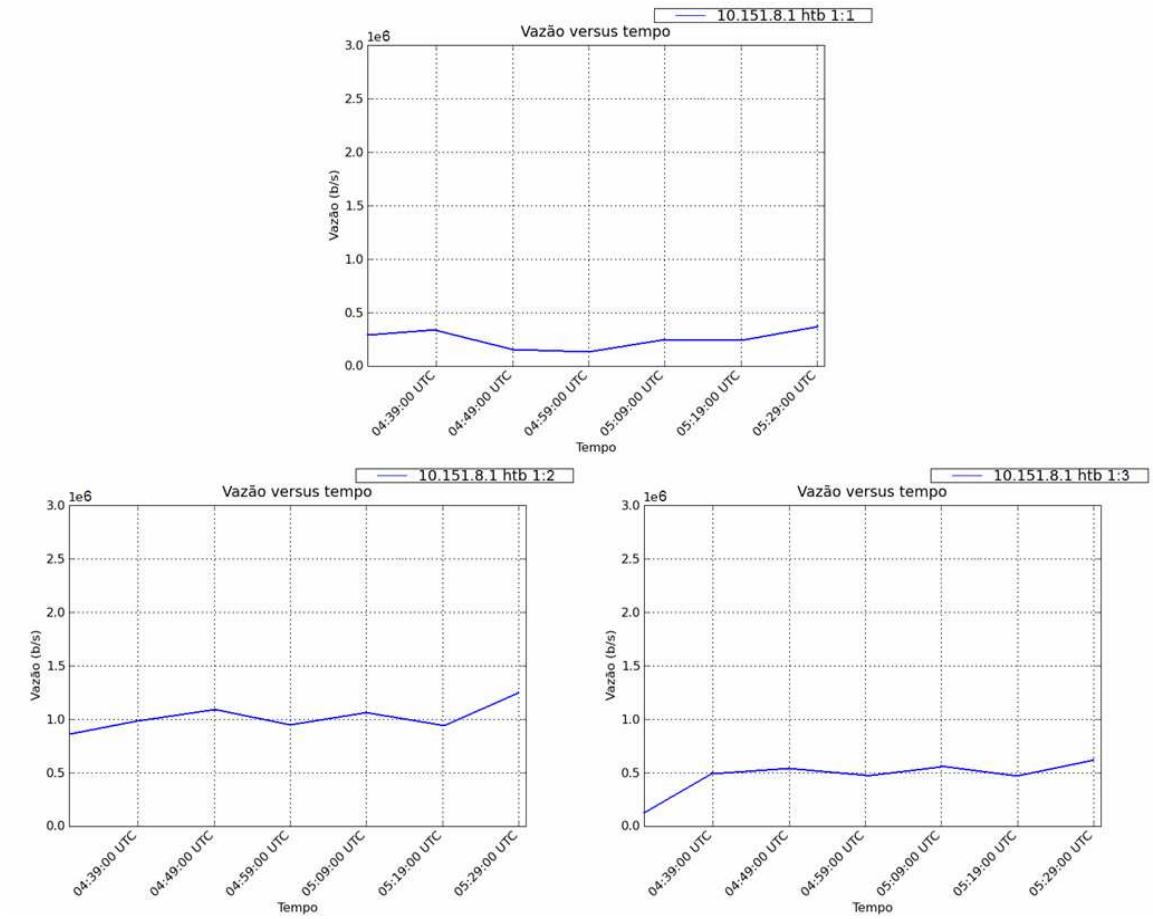

Figura 44. Classes 1:2 e 1:3 usufruindo da banda de 1:1.

Na segunda etapa do teste, o fluxo de dados da classe 1:1 sofreu intencionalmente um aumento até a sua taxa máxima pré-estabelecida (1,5 Mbps) no minuto oitenta, obrigando as classes menos prioritárias 1:2 e 1:3 a devolverem parte da banda que não lhes pertencia, como ilustrado na Figura 44. Assim, a hierarquia de classes do HTB foi obedecida. Com este teste, fica evidente a eficácia da ferramenta *TC MESH* na criação de script*s* com a sintaxe do controle de tráfego GNU/Linux juntamente com a geração de gráficos que auxiliam na visualização tanto em tempo real como de dados passados, permitindo verificar se as políticas de QoS aplicadas atendem as necessidades do usuário.

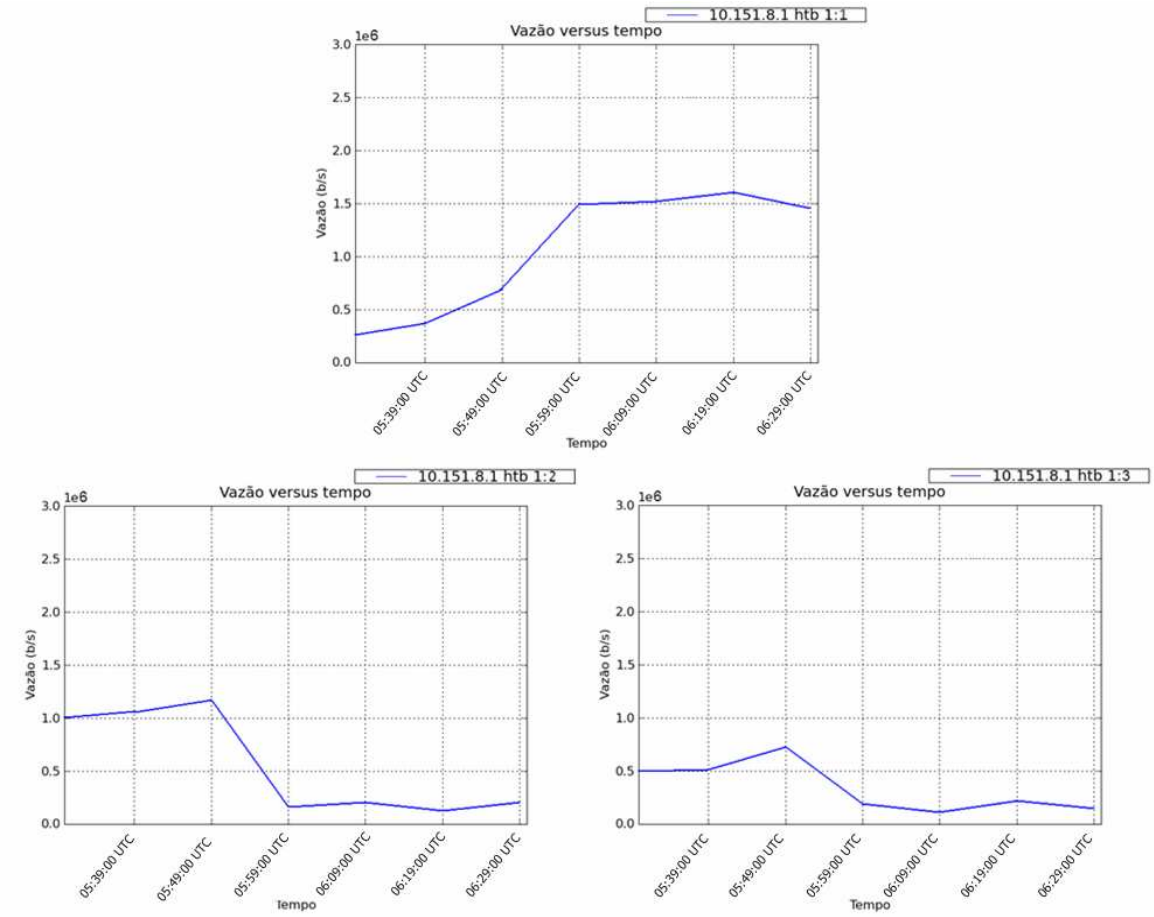

Figura 45. Classe 1:1 recuperando sua banda garantida.

6. CONCLUSÕES

Este trabalho apresentou uma solução para gerência de qualidade de serviço em redes em malha sem fio através da ferramenta *TC MESH*. Com a ferramenta integrada à rede em malha sem fio da UFF tornou-se possível oferecer diferentes níveis de qualidade de serviço de maneira ágil. Com *TC MESH*, foi possível reduzir consideravelmente a complexidade no desenvolvimento de políticas de QoS para redes em malha sem fio, facilitando o uso pelos pesquisadores e estudantes para eventuais testes de QoS na rede bem como otimizar o uso da rede de acordo com as necessidades de cada usuário.

Como o objetivo de uma rede *mesh* é, basicamente, formar um *backbone* sem fio para fornecer acesso faixa larga com custo reduzido, o *TC MESH* provê a escalabidade para a implantação de qualidade de serviço, instalando e executando os scripts em tempo real em todos os roteadores de uma vez e disponibilizando aos usuários diversos níveis de qualidade e a possibilidade da alteração dos mecanismos de diferenciação de serviços de maneira rápida. Dessa forma, é possível montar uma rede de baixo custo para as comunidades menos favorecidas oferecendo vários níveis de qualidade.

Para agregar todas as funcionalidades da ferramenta *TC MESH*, foi incluída uma série de facilidades no que diz respeito a gerência de QoS, entre elas:

- ✓ A criação de uma interface gráfica intuitiva com a possibilidade da especificação de todos os parâmetros de QoS sem o uso de programação.
- ✓ Um modo de configuração alternativo, sem nenhum parâmetro diretamente relacionado ao TC, para usuários que somente têm conhecimentos básicos em redes e nenhum tipo de contato com o controle de tráfego do linux.
- ✓ A possibilidade de criação de disciplinas de enfileiramento dos mais diversos tipos. Desde a tradicional FIFO até o uso do complexo HTB, além da possibilidade de uso do *Diffserv* por meio da disciplina de enfileiramento DSMARK.
- ✓ Um sistema automatizado de instalação e execução das políticas criadas pela ferramenta em todos os roteadores da rede em malha sem fio, facilitando o ajuste das políticas de QoS.

- ✓ Uma ferramenta capaz de coletar dados de cada roteador e o provimento de uma interface gráfica centralizada para geração de gráficos com principais parâmetros de QoS de cada roteador, como a vazão de cada classe e taxa de perdas.
- ✓ O desenvolvimento de um arquivo de configuração XML para permitir a inclusão de novas políticas de QoS para uso no *TC MESH* modo básico, de forma rápida e sem necessidade de programação.

Após a conclusão do desenvolvimento da ferramenta, foi realizado um estudo informal com alunos do laboratório MídiaCom para verificar a aceitação da ferramenta e o ganho médio do tempo (criação, instalação, execução e estatísticas) em relação ao uso do *TC MESH*. Constatou-se que sem o uso do *TC MESH*, um iniciante no uso da ferramenta TC leva em média quatro horas para o desenvolvimento de todos os passos. Já usando a ferramenta *TC MESH*, esse tempo médio é reduzido para aproximadamente quinze minutos. Tal fato foi constatado com três estudantes de mestrado do Departamento de Engenharia de Telecomunicações da UFF.

A principal contribuição do *TC MESH* é a possibilidade de desenvolver políticas de QoS de forma ágil e rápida, sendo esta proposta adaptada para redes em malha sem fio, oferecendo um suporte completo para o usuário gerenciar parâmetros de QoS relevantes à rede, gerenciando em tempo real seu comportamento. Com a facilidade de automatizar o processo de instalação e execução dos scripts, agiliza-se também o trabalho do pesquisador, acelerando a inclusão das políticas em cada roteador.

O oferecimento da ferramenta *TC MESH* na rede *mesh* pode ainda motivar outros pesquisadores a dedicar seus estudos a qualidade de serviço dada a facilidade em testar, analisar e modificar os parâmetros na rede. A ferramenta incentiva mais pessoas a desenvolverem testes com QoS e consequentemente o número de trabalhos desenvolvidos na área.

Além disso, é possível oferecer diversos níveis de serviço, priorizando em especial os serviços em tempo-real como VoIP e videoconferências em redes em malha sem fio, configurando os roteadores de forma ágil de acordo com as necessidades das aplicações em um dado momento.

6.1. TRABALHOS FUTUROS

O *TC MESH* é uma ferramenta que opera na camada de rede do modelo TCP/IP e por isso não é possível obter uma maior precisão na execução das políticas de QoS já que sua operação está restrita às limitações do padrão IEEE 802.11, situado nas camadas MAC e física. Desta forma, um possível trabalho futuro é o mapeamento dos parâmetros de QoS especificados no nível de rede para o nível MAC. Com a integração da ferramenta com a camada MAC, será possível oferecer diversos níveis de serviço de acordo com os parâmetros do padrão IEEE 802.11e, garantindo mais confiabilidade nos níveis de QoS oferecidos.

Outra proposta possível é o desenvolvimento de uma linguagem de alto nível para definição de políticas de QoS, seguindo a proposta apresentada em [TCNG, 2008], que cria uma linguagem mais popular, ou seja, mais conhecida pela comunidade acadêmica. A ferramenta criaria uma linguagem única, independente do TC, para a configuração de diversos tipos de controles de tráfego existentes. Essa linguagem poderia ser mapeada em scripts TC ou em comandos de outra ferramenta para controle de tráfego.

Como proposta futura, também é possível aprimorar a ferramenta de gerência de QoS para que ela se torne mais intuitiva. A proposta seria oferecer ao usuário sugestões de configurações de políticas de acordo com o comportamento da rede e dos tipos de fluxos de pacotes que estão trafegando na rede. Esta proposta desenvolve para o usuário uma política adequada às necessidades de uma rede, cabendo ao usuário a decisão de aceitar ou não a proposta da ferramenta. Um exemplo prático seria uma rede com dois fluxos diferentes concorrentes: um de dados (ftp, por exemplo) e outro multimídia em tempo-real. A ferramenta identificaria esses fluxos oferecendo garantias de vazão para o fluxo de dados, redução do *jitter* e atrasos para o fluxo multimídia.

O desenvolvimento de um banco de dados para armazenar todos os dados estatísticos da rede aprimoriaria a proposta atual. Permitindo ao usuário visualizar não somente os gráficos disponibilizados atualmente pelo TC MESH, mas também criar seus próprios gráficos de acordo com o intervalo de tempo desejado. A integração dos gráficos estatísticos em uma única imagem também é interessante no momento da análise dos dados pelos administradores, permitindo assim a comparação dos valores de cada classe.

Um aspecto importante do *TC MESH* é a ausência de um controle de admissão explícito. Com isso, todos os fluxos possíveis são aceitos para ingressarem na rede até o ponto que o serviço não possa mais ser oferecido. Com isso, a inclusão de uma opção para restringir o número de fluxos entrantes na rede deve ser criada para oferecer com maior precisão níveis

de qualidade para os fluxos da rede. Em [HUBERT, 2009], existem formas de integrar as disciplinas de enfileiramento com técnicas de controle de admissão, policiando todo o tráfego que entra na interface de rede de um equipamento.

7. REFERÊNCIAS BIBLIOGRÁFICAS

ABELÉM, A.; ALBUQUERQUE, C.; SAADE, D.,; AGUIAR, E.; Duarte, J.; DA FONSECA, J.; MAGALHÃES, L., (2007) “Redes Mesh: Mobilidade, Qualidade de Serviço e Comunicação em Grupo”, SBRC2007, Belém.

ACHARYA, A.; MISRA, A.; BANSAL, S. “Challenges in high performance data forwarding in multi -hop wireless networks”, IBM Research Report, Computer Science, Julho de 2002.

ALTQ, Disponível em http://pf4freebsd.love2party.net/altq.html, acessado em abril de 2009.

AWK,“The GNU Awk User`s Guide”, disponível em http://www.gnu.org/software/gawk/manual/gawk.html, acesso em Abril de 2009.

BORTINKOV, E.; VAISMAN, T. K.; **“**A QoS WMN with Mobility Support”, *ACM SIGMOBILE Mobile Computing and Communications Review (MC^2 R)* 12:1, pages 46-48, January 2008, Special Issue on Student Research Competition at ACM Mobicom 2007, Montreal, Canada, Setembro de 2007.

CACTI, The Complete RRDTool-based Graphing Solution, disponível em www.cacti.net, acesso em Maio de 2009.

CARVALHO R.; ABELÉM A.; "phpTCadmin: solução para implementação de qualidade de serviço em redes de computadores baseada em software livre", II Workshop de Tecnologia da Informação das IFES, Gramado-RS 2008.

CHANGKUN W., “PBNM - Policy-Based Network Management”, International Conference on Communication Technology Proceedings, Bejing - China 2000.

CHEN, J.; CARO, A.; MCAULEY, A.; BABA, S.; OHBA, Y.; RAMANATHAN, P. “A QoS Architecture for Future Wireless IP Networks” In Proceedings of the Twelfth IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS), Novembro de 2000.

CHRISTIN N., LIEBEHERR J., The QoSbox: quantitative service differentiation in BSD routers, The International Journal of Computer and Telecommunications Networking, Volume 50 , Paginas: 3353 – 3374, ISSN: 1389-1286, 2006.

CHUN, Brent, CULLER David, ROSCOE, Timothy; PlanetLab: An Overlay Testbed for Broad-Coverage, ACM SIGCOMM Computer Communications Review, Volume 33, Issue 3, 2003.

DUARTE, J. L., PASSOS D., VALLE L. D, OLIVEIRA E., SAADE D. C. M. , ALBUQUERQUE C. V. “Management Issues on Wireless Mesh Networks”, 5th Latin American Network Operations and Management Symposium, Brasil, 2007.

FLOYD S., JACOBSON V., Random early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking, Volume 1, Issue 4, Páginas 397 – 413, ISSN:1063-6692, 1993.

GALLEGO, M.; ALFONSO, D., “Estimação da Capacidade de redes sem fio do tipo Mesh”, Dissertação de Mestrado – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação, Campinas, SP, Novembro de 2005.

GERK, L. F.; “Qualidade de Serviço em Redes sem Fio em Malha”, Relatório Técnico do Laboratório MídiaCom, UFF, Niterói, 2007.

GRANVILLE, L. Z., TAROUCO, M. J. B. ALMEIDA, A. CARISSIMI. 2001. An Approach for integrated management of networks with quality of service support using QAME. In International Workshop on Distributed Systems: Operations and Management, International Workshop on Distributed Systems: Operations and Management, Nancy - France, 1:167-178 INRIA, 2001.

HAMIDIAN, A..; KOLNER, U.,; “Providing QoS in Ad Hoc Networks with Distributed Resource Reservation**”** Managing Traffic Performance in Converged Networks / Lecture Notes in Computer Science, 20th International Teletraffic Congress (ITC20), Ottawa, Canada, Vol. 4516, pp. 309-320, Junho de 2007.

HARTMEIER, D.; “Design and Performance of the OpenBSD Stateful Packet Filter”, Proceedings of the FREENIX Track: 2002 USENIX Annual Technical Conference, 2002.

HIDEKI L., MARTINS R., GUERRANTE A., CARRANO R., MAGALHÃES L. Evaluating the impact of RTS-CTS in OLPC's XOs' Mesh Networks. In: XXV Simpósio Brasileiro de Telecomunicações (SBrT'07), Recife, 2007.

HUBERT B., LARTC - Linux Advanced Routing and Traffic Control, http://lartc.org/, acesso em 2009.

IAN D., ELIZABETH M. Belding-Royer. "AODV Routing Protocol Implementation Design." *Proceedings of the International Workshop on Wireless Ad Hoc Networking (WWAN)*, Tokyo, Japão, Março de 2004.

IANA, www.iana.org, acesso em Março de 2009.

IEEE STANDARTS ASSOCIATION´S BOARD, http://standards.ieee.org/, acesso em 2009.

IPERF. Disponível em: <http://dast.nlanr.net/Projects/Iperf/> Acesso em: 10 março de 2009.

IPFILTER, disponível em http://www.obfuscation.org/ipf/, acesso em Abril de 2009.

IPTABLES, disponível em http://www.netfilter.org/projects/iptables/ index.html, acesso em Abril de 2009.

KUROSE, J. F.; ROSS, K. W. "Redes de Computadores e a Internet - Uma Nova Abordagem", Editora PEARSON Addison Wesley. São Paulo, 2004.

LIEBEHERR, J., CHRISTIN, N., "JoBS: Joint buffer management and scheduling for differentiated services", In Proceedings of IWQoS 2001, pp. 404–418, Karlsruhe, Germany, 2001.

MATPLOTLIB, ://matplotlib.sourceforge.net/, acesso em Março de 2009.

MCKENNEY, Paul E.,Stochastic Fairness Queuing' INFOCOM '90. Ninth Annual Joint Conference of the IEEE Computer and Communication Societies. 'The Multiple Facets of Integration'. Proceedings., IEEE, São Francisco - Califórnia, Junho 1990.

MINIXML, Lightweight XML Library, http://www.minixml.org, acesso em Maio de 2009.

MRTG, The Multi Router Traffic Grapher, http://oss.oetiker.ch/mrtg/, acesso em Maio de 2009.

MUCHALUAT, D. C.; ALBUQUERQUE, C. V.; MAGALHÃES, L. C. S.; PASSOS, D.; DUARTE, J.; VALLE, R., "Redes em Malha: Solução de Baixo Custo para Popularização do Acesso à Internet no Brasil ", *Simpósio Brasileiro de Telecomunicações (SBrT 2007)*, Recife, PE, Brasil, 03 a 06 de setembro de 2007.

OPALSOFT, disponível em: http://opalsoft.net/qos/, acesso em janeiro de 2009.

OPENWRT - http://openwrt.org/, Acesso em 2009.

PEREIRA, T. M.; WESTPHALL, C. B., “Diferenciação de Serviços na Camada de Acesso ao Meio em Redes sem Fio ad hoc” XXV Congresso da Sociedade Brasileira de Computação, São Leopoldo, 2005.

REMESH, Grupo de Pesquisa em Redes Mesh, Universidade Federal Fluminense, http://mesh.ic.uff.br/, 2009.

REMOTE, Rede de Monitoramento de linhas de Transmissão de Energia, disponível em http://mesh.ic.uff.br/, acesso em Março de 2009.

RFC 2474, Definition of the Differentiated Services Field, disponível em www.ietf.org/rfc/rfc2474.txt, acesso em Fevereiro de 2009.

RFC 1157, Simple Network Management *Protocol* (*SNMP*), disponível em www.ietf.org/rfc/rfc2474.txt, acesso em Fevereiro de 2009.

RFC 2205, Resource Reservation Protocol, disponível em http://www.isi.edu/in-notes/rfc2205.txt, acesso em Abril de 2009.

RFC 3317, Differentiated Services Quality of Service Policy Information Base, disponível em ftp://ftp.isi.edu/in-notes/rfc3317.txt, acesso em Maio de 2009.

RRDTOOL, disponível em http://oss.oetiker.ch/rrdtool/, acesso em Março de 2009.

RUBINSTEIN, Marcelo M.; REZENDE, José F.; “Qualidade de Serviço no Controle de Acesso ao Meio de Redes 802.11” Workshop em Qualidade de Serviço e Mobilidade - WQoSM 2002, Angra dos Reis, RJ, Brasil, Novembro 2002.

TCNG, Traffic Control - Next Generation, http://linux-ip.net/gl/tcng/, acesso em Agosto de 2008.

VALLE R., PASSOS D., ALBUQUERQUE C., SAADE D., “Mesh Topology Viewer (MTV): an SVG-Based Interactive Mesh Network Topology Visualization Tool”, *IEEE Symposium on Computers and Communications (ISCC 2008)*, Marrakech, Marrocos, 6 a 9 de julho de 2008.

WRT54GL, “Wireless-G Broadband Router”, disponível em http://www-se.linksys.com/servlet/Satellite?c=L_Product_C2&childpagename=SE%2FLayout&cid=1137451394959&pagename=Linksys%2FCommon%2FVisitorWrapper, acesso em Abril de 2009.

8. ANEXOS

8.1. CÓDIGO-FONTE

```
<?php
// TC-MESH "index.php"
// Desenvolvido por Bruno Lima Wanderley bruwand@midiacom.uff.br
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License as published by
// the Free Software Foundation; either version 2 of the License, or
// (at your option) any later version.
//

session_start();

require_once('common.php');

// zerando subclasses
$_SESSION[classes] = array();
$_SESSION[qdisc_prio] = array();


$iface = NULL;
if (isset($_GET['int'])) {
        $iface = $_GET['int'];
}
if (!isset($int_arr[$iface])) {
        $iface = $int_default;
}


?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>TCMESH - Controle de Trafego para redes Mesh</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<meta name="Description" content="Gerencia de QoS para redes em malha sem
fio" />
<meta name="Keywords" content="qos, mesh, tc" />
<meta name="Author" content="Bruno Wanderley" />
<meta name="Author" content="Allan Denot" />
<link rel="shortcut icon" href="lib/favicon.png" />
<script language="JavaScript" type="text/javascript">


function get_obj (name) {
        return document.getElementById(name);
}

function set_class(arg) {
        select_obj = get_obj('qdisc');
```

```
        if (select_obj.value=="DSMARK") {
                qdisc='qdisc_dsmark=1&';
        } else {
                qdisc = "";
        }

        window.open('set_class.php?'+qdisc+'int='+this.iface+'&view='+this.
view+'&parent='+arg,'Class','width=500,height=450,top=200,left=200');
}

function remove_class(arg) {
        window.open('set_class_post.php?func=remove&id='+arg,'Class','width
=475,height=260,top=200,left=200');
}

function getCheckedValue(radioObj) {
        if(!radioObj)
                return "";
        var radioLength = radioObj.length;
        if(radioLength == undefined)
                if(radioObj.checked)
                        return radioObj.value;
                else
                        return "";
        for(var i = 0; i < radioLength; i++) {
                if(radioObj[i].checked) {
                        return radioObj[i].value;
                }
        }
        return "";
}


function adiciona_classe() {
        var eyeframe;
        eyeframe = document.getElementById('iframe_classes');
        var eyeframedoc;
        eyeframedoc = eyeframe.contentWindow ?
eyeframe.contentWindow.document: eyeframe.contentDocument;
        var classe;
        classesObj = eyeframedoc.getElementById("form_classes").classes;

        classe = getCheckedValue(classesObj);


        set_class(classe);
}

function remove_classe() {
        var eyeframe;
        eyeframe = document.getElementById('iframe_classes');
        var eyeframedoc;
        eyeframedoc = eyeframe.contentWindow ?
eyeframe.contentWindow.document: eyeframe.contentDocument;
        var classe;
        classesObj = eyeframedoc.getElementById("form_classes").classes;

        classe = getCheckedValue(classesObj);

        if (classe == 1) {
                alert("Impossivel Remover Raiz");
                return;
```

```
        }

        remove_class(classe);


}
/*
function show_tc() {
        location = 'get_file.php?set_type=show';
}

function loadfile() {
        window.open('get_file.php?int='+this.iface+'&view='+this.view+'&set
_type=upload','Upload','width=430,height=150,top=200,left=200');
}
*/


function change_qdisc () {

        select_obj = get_obj('qdisc');

        if (select_obj.value=="HTB") {
                get_obj('htb_subclasses_tr').style.display='';
                get_obj('tbf_burst_tr').style.display='none';
                get_obj('tbf_latency_tr').style.display='none';
                get_obj('tbf_peakrate_tr').style.display='none';
                get_obj('tbf_minburst_tr').style.display='none';
                get_obj('sfq_perturb_tr').style.display='none';
                get_obj('dsmark_indices_tr').style.display='none';
                get_obj('dsmark_default_index_tr').style.display='none';
                get_obj('dsmark_tos_tr').style.display='none';
                get_obj('prio_filters_tr').style.display='none';

        } else if (select_obj.value=="TBF") {
                get_obj('htb_subclasses_tr').style.display='none';
                get_obj('tbf_burst_tr').style.display='';
                get_obj('tbf_latency_tr').style.display='';
                get_obj('tbf_peakrate_tr').style.display='';
                get_obj('tbf_minburst_tr').style.display='';
                get_obj('sfq_perturb_tr').style.display='none';
                get_obj('dsmark_indices_tr').style.display='none';
                get_obj('dsmark_default_index_tr').style.display='none';
                get_obj('dsmark_tos_tr').style.display='none';
                get_obj('prio_filters_tr').style.display='none';

        } else if (select_obj.value=="SFQ") {
                get_obj('htb_subclasses_tr').style.display='none';
                get_obj('tbf_burst_tr').style.display='none';
                get_obj('tbf_latency_tr').style.display='none';
                get_obj('tbf_peakrate_tr').style.display='none';
                get_obj('tbf_minburst_tr').style.display='none';
                get_obj('sfq_perturb_tr').style.display='';
                get_obj('dsmark_indices_tr').style.display='none';
                get_obj('dsmark_default_index_tr').style.display='none';
                get_obj('dsmark_tos_tr').style.display='none';
                get_obj('prio_filters_tr').style.display='none';

        } else if (select_obj.value=="DSMARK") {
                get_obj('htb_subclasses_tr').style.display='';
                get_obj('tbf_burst_tr').style.display='none';
                get_obj('tbf_latency_tr').style.display='none';
```

```
                get_obj('tbf_peakrate_tr').style.display='none';
                get_obj('tbf_minburst_tr').style.display='none';
                get_obj('sfq_perturb_tr').style.display='none';
                get_obj('dsmark_indices_tr').style.display='';
                get_obj('dsmark_default_index_tr').style.display='';
                get_obj('dsmark_tos_tr').style.display='';
                get_obj('prio_filters_tr').style.display='none';

        } else if (select_obj.value=="PRIO") {
                get_obj('htb_subclasses_tr').style.display='none';
                get_obj('tbf_burst_tr').style.display='none';
                get_obj('tbf_latency_tr').style.display='none';
                get_obj('tbf_peakrate_tr').style.display='none';
                get_obj('tbf_minburst_tr').style.display='none';
                get_obj('sfq_perturb_tr').style.display='none';
                get_obj('dsmark_indices_tr').style.display='none';
                get_obj('dsmark_default_index_tr').style.display='none';
                get_obj('dsmark_tos_tr').style.display='none';
                get_obj('prio_filters_tr').style.display='';
        }
}

function set_prio (prio) {
        // prio = high|mid|low
        window.open('set_class.php?qdisc_prio='+prio+'&int='+this.iface+'&p
arent=1','Class','width=500,height=450,top=200,left=200');
}

</script>
<link rel="stylesheet" href="style.css" type="text/css" />
</head>

<body onload="change_qdisc();">
<form method="post" action="savefile.php" target="_new">
<table align="center" cellpadding=0 border=0 width="700">
        <tr>
                <td align="right"><img src="logo_midiacom.jpg" alt=""
width="170" height="80" /></td>
                <td align="left" class="title_cell">
                        <span style="font-weight:bold;font-size:28px;">TC
MESH</span><br />
                        <span style="font-size:14px;">Ger&ecirc;ncia de QoS
para redes em malha</span>
                </td>
        </tr>
</table>
<p />
<table align="center" cellpadding=0 cellspacing=0 border=0 width="760">
        <tr>
                <td class="main_form_caption_cell">Interface</td>
                <td class="main_form_value_cell"><?php echo
cria_select('interface', $int_arr, $iface,""); ?></td>
        </tr>
        <tr>
                <td class="main_form_caption_cell">Velocidade da
Interface</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" name="maxspeed" size="4" /> Mbps</td>
        </tr>
        <tr>
                <td class="main_form_caption_cell">Disciplina de
Enfileiramento</td>
```

```
                <td class="main_form_value_cell">
                        <select id="qdisc" onchange="change_qdisc();"
name="qdisc">
                                <option selected="selected">HTB</option>
                                <option>TBF</option>
                                <option>SFQ</option>
                                <option>DSMARK</option>
                                <option>PRIO</option>
                        </select>
                </td>
        </tr>
        <tr id="tbf_burst_tr" style="display:none;">
                <td class="main_form_caption_cell">Burst</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" value="5" name="burst" size="4" /> kb</td>
        </tr>
        <tr id="tbf_latency_tr" style="display:none;">
                <td class="main_form_caption_cell">Latency</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" name="latency" value="1" size="4" /> ms</td>
        </tr>
        <tr id="tbf_peakrate_tr" style="display:none;">
                <td class="main_form_caption_cell">Peakrate</td>
                <td class="main_form_value_cell">
                        <input type="text" class="main_form_textbox"
value="1" name="peakrate" size="4" />
                        <select name="peakrate_tax">
                                <option value="Mbit"
selected="selected">Mbits/s</option>
                                <option value="Kbit">Kbits/s</option>
                                <option value="bit">bits/s</option>
                        </select>
                </td>
        </tr>
        <tr id="tbf_minburst_tr" style="display:none;">
                <td class="main_form_caption_cell">Latency</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" value="1540" name="minburst" size="4" /> ms</td>
        </tr>
        <tr id="sfq_perturb_tr" style="display:none;">
                <td class="main_form_caption_cell">Perturb</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" value="15" name="perturb" size="4" /></td>
        </tr>

        <tr id="dsmark_indices_tr" style="display:none;">
                <td class="main_form_caption_cell">Indices</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" value="8" name="indices" size="4" /></td>
        </tr>
        <tr id="dsmark_default_index_tr" style="display:none;">
                <td class="main_form_caption_cell">Default Index</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" value="8" name="default_index" size="4" /></td>
        </tr>
        <tr id="dsmark_tos_tr" style="display:none;">
                <td class="main_form_caption_cell">TOS</td>
                <td class="main_form_value_cell"><input type="text"
class="main_form_textbox" value="28" name="tos" size="4" /></td>
        </tr>
        <tr id="prio_filters_tr" style="display:none;">
                <td class="main_form_caption_cell">Prioridades</td>
```

```
                <td class="main_form_value_cell">
                        <input type="button" value="Alta"
onclick="set_prio('high');" />
                        <input type="button" value="Média"
onclick="set_prio('mid');" />
                        <input type="button" value="Baixa"
onclick="set_prio('low');" /><br />
                        <iframe id="iframe_classes_prio" frameborder="0"
width="560" height="100" src="classes_prio.php"></iframe>
                </td>
        </tr>
        <tr id="htb_subclasses_tr">
                <td class="main_form_caption_cell">Subclasses</td>
                <td class="main_form_value_cell" style="text-align: right;">
                        <iframe id="iframe_classes" frameborder="0"
width="560" height="200" src="classes.php"></iframe> <br/>
                        <input type="button" value="Adicionar classe/filtro"
class="main_form_botao" onclick="adiciona_classe();" />
                        <input type="button" value="Remover"
class="main_form_botao" onclick="remove_classe();" />
                </td>
        </tr>
        <tr>
                <td style="border-bottom: 0px;"
class="main_form_caption_cell">Script</td>
                <td style="border-bottom: 0px;"
class="main_form_value_cell">
                        <input type="submit" name="savefile" value="Salvar"
class="main_form_botao" />
                        <input type="submit" name="openfile" value="Abrir
Novo" class="main_form_botao" />
                </td>
        </tr>
</table>
</form>
</body>
</html>
```

```
<?php
```

```
session_start();

$classes = $_SESSION[classes];
if (!$classes) {
	$classes = array();
}

if ($_GET[remove]) {
	foreach ($classes as $key => $value) {
		if ($value[id]==$_GET[remove]) {
			unset($classes[$key]);
		}
	}
	$_SESSION[classes]=$classes;
}

//print_r($_SESSION[classes]);


?>
<HTML>
<HEAD>
<STYLE>
label {
	font-family: Verdana,Helvetica;
	font-size: 12px;
}
</STYLE>
</HEAD>
<BODY>
<form id="form_classes">
<table cellpadding="3" cellspacing="0" border="0" width="100%">

	<tr>
		<td><input id="root" type="radio" name="classes" value="1"
checked="checked" /><label for="root">Raiz</label></td>
	</tr>

	<?php
	function imprime_classes($classes,$nivel=1,$pais="") {

		foreach ($classes as $id => $class) {
			if (!is_array($class))
				continue;

			$lever =
str_repeat("     ",$nivel);
	?>
	<tr>
		<td>
			<?=$lever?><input id="<?=$pais.$id?>" type="radio"
name="classes" value="<?=$pais.$id?>" />
			<label
for="<?=$pais.$id?>"><?=$class[name]?></label>
		</td>
	</tr>
	<?php
			if (count($class[children])>0) {
				// faz a recursao e chama de novo com um
nivel mais alto
```

```
        imprime_classes($class[children],$nivel+1,$pais.$id);
                        }

                }
        }

        imprime_classes($_SESSION[classes]);
        ?>

</table>
</form>
</BODY></HTML>
```

```
<?php

session_start();
```

```
$classes = $_SESSION[qdisc_prio];


if ($_SESSION[qdisc_prio][high])
        $high = $_SESSION[qdisc_prio][high][name];
else
        $high = "Não definida.";
if ($_SESSION[qdisc_prio][mid])
        $mid = $_SESSION[qdisc_prio][mid][name];
else
        $mid = "Não definida.";
if ($_SESSION[qdisc_prio][low])
        $low = $_SESSION[qdisc_prio][low][name];
else
        $low = "Não definida.";

?>
<HTML>
<HEAD>
<STYLE>
td {
        font-family: Verdana,Helvetica;
        font-size: 12px;
}
</STYLE>
</HEAD>
<BODY>
<table cellpadding="3" cellspacing="0" border="0">

        <tr>
                <td><strong>Alta: </strong></td><td><?=$high?></td>
        </tr>
        <tr>
                <td><strong>Média: </strong></td><td><?=$mid?></td>
        </tr>
        <tr>
                <td><strong>Baixa: </strong></td><td><?=$low?></td>
        </tr>
</table>
</BODY></HTML>
```

```
<?php

set_time_limit(0);
ignore_user_abort(true);

include "common.php";
```

```
$script = stripslashes($_POST[script]);

if ($_POST[download]) {

        header("Content-Disposition: attachment; filename=\"tcmesh.sh\"");
        header("Content-Type: application/force-download");
        header("Content-Length: ".strlen($script));
        echo $script;
        exit();
}

?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>TCMESH - Controle de Trafego para redes Mesh</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<meta name="Description" content="Gerencia de QoS para redes em malha sem
fio" />
<meta name="Keywords" content="qos, mesh, tc" />
<meta name="Author" content="Bruno Wanderley" />
<meta name="Author" content="Allan Denot" />
<link rel="shortcut icon" href="lib/favicon.png" />
<link rel="stylesheet" href="style.css" type="text/css" />
<style>

.mensagem {
        font-family: Verdana,Helvetica;
        font-size: 12px;
        font-weight: bold;
        float: left;
        padding: 4px;
}
.status {
        font-family: Verdana,Helvetica;
        font-size: 12px;
        font-weight: bold;
        float: right;
        padding: 4px;
        clear:right;
        color: red;
        position: absolute;
        right:0px;
        background-color: white;
}

.codigo {
        font-family: 'Courier New';
        font-size: 12px;
        padding: 4px 0px 4px 14px;
        clear:left;
        width: 90%;
}




</style>
<script language='javascript'>
function hide_load() {
        document.getElementById('loading').style.display='none';
}
```

```
</script>
</head>
<body>
<?php

function imprime_mensagem ($mensagem) {
        echo "<div class='mensagem'>$mensagem</div>\n";

        echo "<div id='loading' class='status'><img width=19 height=18
src='loading.gif' /></div>\n";

        flush();
}
function imprime_status ($status) {
        global $codigo;
        $img_ok = "<img src='ok.gif' />";

        if ($status==true) {
                $status = $img_ok;
        } else {
                $status = "FALHA";
                $exit_on_end=true;
        }
        //echo "<script
language='javascript'>alert('oi');hide_load();</script>\n";

        echo "<div class='status'>$status</div>\n";

        echo "<div class='codigo'>".nl2br($codigo)."</div>\n";
        flush();
        $codigo = "";

        flush();

        if ($exit_on_end)
                exit();

}
function imprime_codigo ($cod) {
        global $codigo;
        $codigo .= $cod;
}

if ($_POST[execute]) {

        $ip_gateway = "200.20.15.72";

        imprime_mensagem("Criando Arquivo");

        if (!DUMMY)
imprime_status(file_put_contents("/tmp/tcmesh.sh",$script));
        if (DUMMY) imprime_status(true);

        imprime_mensagem("Enviando para o gateway");

        if (!DUMMY) imprime_codigo(shell_exec(' sudo /usr/bin/scp
/tmp/tcmesh.sh root@'.$ip_gateway.':/tmp/tcmesh.sh'));
        if (!DUMMY) imprime_codigo(shell_exec(' sudo /usr/bin/ssh
root@'.$ip_gateway.' "chmod +x /tmp/tcmesh.sh"'));
        if (!DUMMY) imprime_codigo(shell_exec(' sudo /usr/bin/ssh
root@'.$ip_gateway.' "/tmp/tcmesh.sh"'));
```

```
        if (DUMMY) imprime_codigo("ajcndkjc ajc ndajkn
cd\ncadndajkndc\nncdjnckjd..");

        sleep(5);
        imprime_status(true);

        if ($_POST[remote]=="ip") {
                $ip = $_POST[ip];
                if (!validate_ip_address($ip)) {
                        imprime_mensagem("Endereço IP inválido");
                        imprime_status(false);
                }

                imprime_mensagem("Copiando para o roteador $ip");
                if (!DUMMY) imprime_codigo(shell_exec(' sudo ssh
root@'.$ip_gateway.' "scp -i /etc/dropbear/dropbear_rsa_host_key
/tmp/tcmesh.sh '.$ip.':/tmp/tcmesh.sh'));
                if (DUMMY) imprime_codigo("ajcndkjc ajc ndajkn
cd\ncadndajkndc\nncdjnckjd..");
                sleep(5);
                imprime_status(true);

                imprime_mensagem( "Executando");;
                if (!DUMMY) imprime_codigo(shell_exec(' sudo ssh
root@'.$ip_gateway.' "ssh -i /etc/dropbear/dropbear_rsa_host_key
root@'.$ip.' \'ash /tmp/tcmesh.sh\'"'));
                if (DUMMY) imprime_codigo("ajcndkjc ajc ndajkn
cd\ncadndajkndc\nncdjnckjd..");
                imprime_status(true);

        } else {

                imprime_mensagem( "Copiando para todos em broadcast");
                if (!DUMMY) imprime_codigo(shell_exec('/usr/bin/ssh -i
/var/id_rsa root@'.$ip_gateway.' "bcp /tmp/tcmesh.sh /tmp/tcmesh.sh"'));
                sleep(5);
                imprime_status(true);

                imprime_mensagem( "Executando");
                if (!DUMMY) imprime_codigo(shell_exec('sudo /usr/bin/ssh -i
/var/id_rsa root@'.$ip_gateway.' "bshell \'ash /tmp/tcmesh.sh\'"'));
                imprime_status(true);
        }

        imprime_mensagem( "Processo de copia e execucao finalizados");
        imprime_status(true);

}

?>


</body>
</html>
<?php

session_start();

include "common.php";

//print_r($_SESSION[classes]);
```

```
// campos recebidos:
// interface / maxspeed / SESSION[classes]

if ($_POST[openfile]) {

        $script = "";


} else if ($_POST[savefile]||$_POST[savefile_pc]) {
        $defs = loadxml("tc.xml");


        // variaveis vindas do formulario comuns a todos os QDisc
        $def['interface'] = $_POST['interface'];
        $def[maxspeed] = $_POST[maxspeed];
        $def[data] = date("D M j H:i:s Y");

        $log = "";

        // carregando os comandos de filtros
        $array_subfilters_name = array();
        $array_subfilters_command = array();
        foreach ($defs[tc][filters][filter] as $filter_id => $filter) {
                if (!$filter[name]) { continue; }

                for
($subfilter_id=0;$subfilter_id<$filter[subfilter][_num];$subfilter_id++) {
                        $array_subfilters_name[$filter_id.$subfilter_id]     = $filter
                        $array_subfilters_command[$filter_id.$subfilter_id]
= $filter[subfilter][$subfilter_id][command];
                }
        }

        $log .= "Inicializando criação do script TC MESH em
".$def[data].".\n";

        // primeiro passo:
        //              inicializacao do script
        if ($_POST[savefile]) {
                $tpl_init = $defs[tc][init];
                $log .= "Versão para Roteadores utilizada\n";
        }
        else if($_POST[savefile_pc]) {
                $tpl_init = $defs[tc]['init-pc'];
                $log .= "Versão para PC utilizada\n";
        }

        $script = tpl_replace($def,$tpl_init)."\n\n";

        // carregando templates para tamanho e u32
        $tpl_size = $defs[tc][size]; // vars: -
        $tpl_u32 = $defs[tc][u32];

        $log .= "Interface selecionada: ". $def['interface']."\n";
        $log .= "Velocidade máxima da interface selecionada:
".$def[maxspeed]."\n";
        $log .= "Criando classes/filtros\n";

        // funcao utilizada apenas para o QDisc HTB
        function load_classes ($def_global,$classes,$parent_id=1) {
                global $tpl_subclass;global $tpl_u32;global $tpl_size;
                global $array_subfilters_name;
```

```
		global $array_subfilters_command;
		global $log;
		$fwmark = 10; // comeca de 11


		foreach ($classes as $class_id => $class) {
			if (!$class[name]) continue;

			$fwmark++;

			// o ID da classe vai ser: $class[parent].$class_id


			$def = array();

			$def[prio] = $class[htb_prio];
			$def[id] = $parent_id.$class_id;

			$def[parent]           = $parent_id;
			$def[htb_rate]         = $class[htb_rate];
			$def[htb_tax]          = $class[htb_tax];
			$def[htb_max]          = $class[htb_max];
			$def[htb_max_tax]      = $class[htb_max_tax];
			if ($class[ceil]==1)
				$def[ceil] = "ceil";
			else
				$def[ceil] = "";

			$def[fwmark]           = $fwmark;

			// juntando as definicoes globais com as locais, pra
na hora de fazer o replace
			$def = array_merge($def,$def_global);

			$script .= "# Subclasse ".$def[id]."\n";
			$script .= tpl_replace($def,$tpl_subclass)."\n\n";

			$name = $class[name];

			if ($class[mode]=="basic") {

				//$name                =
$array_subfilters_name[$class[filter].$class[subfilter]];
				$tpl_filter =
$array_subfilters_command[$class[filter].$class[subfilter]]; // variaveis:
interface|id|prio


				$script .= "# Filtro ".$def[id]." - $name\n";
				$script .=
tpl_replace($def,$tpl_filter)."\n\n";

			} else if ($class[mode]=="advanced") {

				if ($class[filtertype]=="u32") {
					$u32 = "";$array_u32=array();
					// ip_src / ip_dst / port_src /
port_dst / tos / protocol

					if ($class[ip_src])
						$array_u32[] = "src
".$class[ip_src];
```

```
                                        if ($class[ip_dst])
                                                $array_u32[] = "dst
".$class[ip_dst];

                                        if ($class[port_src])
                                                $array_u32[] = "sport
".$class[port_src];

                                        if ($class[port_dst])
                                                $array_u32[] = "dport
".$class[port_dst];

                                        if ($class[tos])
                                                $array_u32[] = "tos
".$class[tos];

                                        if (intval($class[protocol])!=0)
                                                $array_u32[] = "protocolo
".$class[protocol];

                                        $def[u32] = implode (" ",$array_u32);

                                        $script .= "# Filtro U32
".$def[id]."\n";
                                        $script .=
tpl_replace($def,$tpl_u32)."\n\n";


                                } else if ($class[filtertype]=="size") {
                                        // size_min / size_max

                                        $def['size_min'] =
$class['size_min'];
                                        $def['size_max'] =
$class['size_max'];

                                        $script .= "# Filtro de Tamanho do
Pacote ".$def[id]." - $name\n";
                                        $script .=
tpl_replace($def,$tpl_size)."\n\n";
                                }

                                $log .= "  Subclasse ID: ".$def[id]."
criada\n";
                                $log .= "    Taxa:
".$def[htb_rate].$def[htb_tax]."\n";
                                $log .= "    Taxa Máxima:
".$def[htb_max].$def[htb_max_tax]."\n";
                                $log .= "    Emprestar?: ".($class[ceil] ?
"Sim" : "Não")."\n";

                                $log .= "  Filtro ID: ".$def[id]." criado\n";
                                $log .= "    Nome: '$name'\n";

                        }

                        // faz a chamada recursivamente, se houver filhos...
                        if (count($class[children])>0)
                                $script .=
load_classes($def_global,$class[children],$def[id]);
```

```
                }
                return $script;
        }

        if ($_POST[qdisc]=="HTB") {

                $classes = $_SESSION[classes];

                // carrega os templates
                $tpl_subclass = $defs[tc][subclass]; // variaveis:
parent|id|htb_rate|htb_tax|htb_max|htb_max_tax|ceil
                $tpl_htb = $defs[tc][qdisc][htb];

                $script .= tpl_replace($def,$tpl_htb);

                $script .= load_classes($def,$classes);

        } else if ($_POST[qdisc]=="TBF") {
                // carrega template
                $tpl_tbf = $defs[tc][qdisc][tbf]; // vars: burst / latency /
peakrate / peakrate_tax / minburst

                // seta variaveis
                $def[burst]                     = $_POST[burst];
                $def[latency]           = $_POST[latency];
                $def[peakrate]          = $_POST[peakrate];
                $def[peakrate_tax]      = $_POST[peakrate_tax];
                $def[minburst]          = $_POST[minburst];

                $script .= tpl_replace($def,$tpl_tbf);

        } else if ($_POST[qdisc]=="SFQ") {
                // carrega template
                $tpl_sfq = $defs[tc][qdisc][sfq]; // vars: perturb

                // seta variaveis
                $def[perturb]  = $_POST[perturb];

                $script .= tpl_replace($def,$tpl_sfq);

        } else if ($_POST[qdisc]=="DSMARK") {
                // carrega template
                $tpl_dsmark = $defs[tc][qdisc][dsmark]; // vars: indices /
default_index / tos
                $tpl_subclass = $defs[tc][subclass]; // variaveis:
parent|id|htb_rate|htb_tax|htb_max|htb_max_tax|ceil

                // seta variaveis
                $def[indices]           = $_POST[indices];
                $def[default_index]     = $_POST[default_index];
                $def[tos]                       = $_POST[tos];

                $script .= tpl_replace($def,$tpl_dsmark);

                // carrega classes (somente primeiro nivel)
                $classes = $_SESSION[classes];

                $script .= load_classes($def,$classes);


        } else if ($_POST[qdisc]=="PRIO") {
                // carrega template
```

```
                $tpl_prio = $defs[tc][qdisc][prio]; // vars: -
                // nao vou carregar o template do subclass, pq aqui nao há
subclasses, somente filtros


                // vou copiar os 3 filtros de prioridade pro array de
classes pra poder chamar a funcao load_classes acima.
                $classes[] = $_SESSION[qdisc_prio][high];
                $classes[][htb_prio] = 1;

                $classes[] = $_SESSION[qdisc_prio][mid];
                $classes[][htb_prio] = 2;

                $classes[] = $_SESSION[qdisc_prio][low];
                $classes[][htb_prio] = 3;

                $script .= tpl_replace($def,$tpl_prio);

                $script .= load_classes($def,$classes,"");

        }

        $log .= "Fim geração do Script";



}


?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>TC MESH: Abrir/Salvar Script</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<meta name="Description" content="TC MESH" />
<meta name="Keywords" content="qos, tc" />
<meta name="Author" content="Bruno Wanderley" />
<meta name="Author" content="Allan Denot" />
<link rel="shortcut icon" href="lib/favicon.png" />
<link rel="stylesheet" href="style.css" type="text/css" />
<script language="JavaScript" type="text/javascript">

</script>
</head>
<body>
<table align="center" cellpadding=0 cellspacing=0 border=0 width="700">
        <tr>
                <td align="left"><span align="left" class="title_cell"
style="font-size:24px; font-weight:bold;">Abrir/Salvar Script</span></td>
        </tr>
</table>
<form method="post" action="savefile.php">
<input type="hidden" name="interface" value="<?=$_POST['interface'];?>" />
<input type="hidden" name="maxspeed" value="<?=$_POST['maxspeed'];?>" />
<input type="hidden" name="qdisc" value="<?=$_POST['qdisc'];?>" />

<table style="margin-top:8px;" align="center" cellpadding=0 cellspacing=0
border=0 width="700">
        <tr>
```

```
                <td style="padding: 2px;font-family: Verdana,Helvetica;font-
size:12px;" bgcolor="#ddd">Logs</td>
        </tr>
        <tr>
                <td>
                        <textarea style="width:700px;" name="log"
readonly="readonly"
                                id="script" cols="100" rows="6"><?php echo
$log; ?></textarea>
                </td>
        </tr>
        <tr>
                <td style="padding: 2px;font-family: Verdana,Helvetica;font-
size:12px;" bgcolor="#ddd">Script</td>
        </tr>
        <tr>
                <td>
                        <textarea style="width:700px;" name="script"
id="script"
                                cols="100" rows="20"><?php echo $script;
?></textarea>
                </td>
        </tr>
        <tr>
                <td align="center">

<?php if ($_POST[savefile]) { ?>
<input type="submit" name="savefile_pc" value="Versão para PC" />
<?php } else if ($_POST[savefile_pc]) { ?>
<input type="submit" name="savefile" value="Versão para Roteadores" />
<?php } ?>

                </td>
        </tr>
</table>

</form>



<form action="execute.php" method="post" target="iframe_execute"
        onsubmit="
                document.getElementById('hidden_script').value =
document.getElementById('script').value;
                "
>
<p align="center"><input type="submit" name="download" value="Download"
/></p><br />
<input type="hidden" name="script" id="hidden_script" />

<?php if ($_POST[savefile]||$_POST[openfile]) { ?>

<table align="center" cellpadding=0 cellspacing=0 border=0 width="700">
        <tr>
                <td align="left" colspan="2"><span align="left"
class="title_cell" style="font-size:18px; font-weight:bold;">Executar
Remotamente</span></td>
        </tr>
        <tr>
                <td class="main_form_caption_cell">Endereço IP</td>
                <td class="main_form_value_cell">
```

```
                        <input type="radio" name="remote" value="ip"
checked="checked" id="remote_ip" />
                        <select name="ip"
onfocus="document.getElementById('remote_ip').checked='true';">
                                <option>10.151.1.1</option>
                                <option>10.151.5.1</option>
                                <option>10.151.17.1</option>
                                <option>10.151.18.1</option>
                                <option>10.151.23.1</option>
                                <option>10.151.26.1</option>
                                <option>10.151.27.1</option>
                                <option>10.151.100.1</option>
                        </select>

                        <!--
                        18 , 17 , 26 , 5 , 27 , 1 , 23 , 100
                        <input type="text" name="ip" size="18"
maxlength="20" class="text"

        onfocus="document.getElementById('remote_ip').checked='true';" />
                                -->
                </td>
        </tr>
        <tr>
                <td class="main_form_caption_cell"></td>
                <td class="main_form_value_cell">
                        <input type="radio" name="remote" value="broadcast"
id="remote_broadcast" />
                        <label for="remote_broadcast">Broadcast</label>
                </td>
        </tr>
        <tr>
                <td style="border-bottom: 0px;"
class="main_form_caption_cell"> </td>
                <td style="border-bottom: 0px;"
class="main_form_value_cell">
                        <input type="submit" name="execute" value="Executar"
class="submit"

        onclick="document.getElementById('iframe_execute').style.display=''
;";
                         />
                </td>
        </tr>
</table>

<?php } ?>

</form>
<p align="center">
<iframe style="display:none;" id="iframe_execute" name="iframe_execute"
frameborder="0" width="700" height="250" src="execute.php"></iframe>
</p>

</body>
</html>
```

```
<?php

session_start();

include "common.php";



if ($_POST[func]=="set_class") {

        // carregando filtros do XML
        list($array_filters,$array_subfilters) = load_filters();


        $filter = $_POST[filter];
        $subfilter = $_POST["subfilter-".$filter];

        $parent = $_POST['parent'];

        $class['parent']=$parent;

        // htb_rate / htb_tax / htb_max_tax / htb_max / htb_prio

        $class['htb_rate']            = $_POST['htb_rate'];
        $class['htb_max_tax']  = $_POST['htb_max_tax'];
        $class['htb_tax']             = $_POST['htb_tax'];
        $class['htb_max']             = $_POST['htb_max'];
        $class['htb_prio']            = $_POST['htb_prio'];
        $class['ceil']                = $_POST['ceil'];
        $class['children']            = array();

        $class[mode]                  = $_POST[mode];


        if ($_POST[mode]=="basic") {

                // modo basico: filter / subfilter
                $class['filter']=$filter;
                $class['subfilter']=$subfilter;


                $class['name'] = $array_filters[$filter]." /
".$array_subfilters[$filter][$subfilter].
                                                " /
".$class[htb_rate].$class[htb_tax].
                                                " / máx:
".$class[htb_max].$class[htb_max_tax].
                                                " / prio: ".$class[htb_prio].
                                                ($class[ceil]==1 ? " / ceil"
: null);


        } else if ($_POST[mode]=="advanced") {

                if ($_POST['filtertype']=="size") {
                        if ($_POST['size_min']=="" ||
$_POST['size_max']=="") {
                                die ("Você deve entrar com um valor mínimo e
máximo.");
```

```
                        }
                        if
(intval($_POST['size_min'])>=intval($_POST['size_max'])) {
                                die ("O valor do tamanho mínimo do pacote
deve ser menor que o tamanho máximo.");
                        }
                }


                // modo avancado: ip_src / ip_dst / port_src / port_dst /
size_max | size_min | tos
                $class['ip_src']              = $_POST['ip_src'];
                $class['ip_dst']              = $_POST['ip_dst'];
                $class['port_src']            = $_POST['port_src'];
                $class['port_dst']            = $_POST['port_dst'];
                $class['size_max']            = $_POST['size_max'];
                $class['size_min']            = $_POST['size_min'];
                $class['tos']                 = $_POST['tos'];
                $class['protocol']            = $_POST['protocol'];
                $class['filtertype']   = $_POST['filtertype'];

                $class['name'] =
                                ($class[ip_src]||$class[port_src] ? "src:
".$class[ip_src].":".$class[port_src]." / " : null).
                                ($class[ip_dst]||$class[port_dst] ? "dst:
".$class[ip_dst].":".$class[port_dst]." / " : null).
                                ($class[size_max]||$class[size_min] ? "size:
".$class[size_min]."-".$class[size_max]." / " : null).
                                ($class[tos] ? "tos: ".$class[tos]." / " :
null).
                                $class[htb_rate].$class[htb_tax]." / ".
                                "máx: ".$class[htb_max].$class[htb_max_tax]."
/ ".
                                "prio: ".$class[htb_prio].
                                ($class[ceil] ? " / ceil" : null);

        }

        if ($_POST[qdisc_prio]!="") {
                // no caso de PRIO, eh um array com 3 chaves: high/mid/low

                $class['parent']="";

                $_SESSION[qdisc_prio][$_POST[qdisc_prio]]=$class;

        } else if ($parent == 1) {
                // os outros casos sao para HTB, vai colocar a classe (e
filtro) criados em um array em formato de arvore

                // pai eh a raiz, entao insere na raiz.
                if (!$_SESSION[classes]) {
                        // crio 2 dummies soh para comecar do 2 nesse caso.
                        $_SESSION[classes][0]=1;
                        $_SESSION[classes][]=1;
                }
                $_SESSION[classes][] = $class;
        } else {
                // o pai eh outro, entao preciso achar e colocar como
children

                // =& eh referencia, entao vou descendo o nivel da
referencia ateh achar o que eu quero,
```

```
                // depois basta adicionar o children
                $tmp =& $_SESSION[classes];
                for ($c=0;$c<strlen($parent);$c++) {
                        $tmp =& $tmp[$parent[$c]][children];
                }

                $tmp[]=$class;

        }


} else if ($_GET[func]=="remove") {
        $id = $_GET[id];

        echo "removendo $id";

        $tmp =& $_SESSION[classes];
        for ($c=0;$c<strlen($id);$c++) {
                if ($c==strlen($id)-1) {
                        unset($tmp[$id[$c]]);
                        break;
                }
                $tmp =& $tmp[$id[$c]][children];
        }

        //print_r($tmp);

}
?><HTML>
<HEAD>
<SCRIPT language="javascript">
var eyeframe;

<?php if ($_POST[qdisc_prio]!="") { ?>
eyeframe = opener.document.getElementById('iframe_classes_prio');
<?php } else { ?>
eyeframe = opener.document.getElementById('iframe_classes');
<?php } ?>


var eyeframedoc;
eyeframedoc = eyeframe.contentWindow ? eyeframe.contentWindow.document:
eyeframe.contentDocument;
eyeframedoc.location.reload();

if (eyeframedoc.getElementById("root")) {
        eyeframedoc.getElementById("root").checked='true';
}

self.close();
</SCRIPT>
</HEAD>
<BODY>
OK
</BODY>
</HTML>
```

8.2. ARQUIVO DE CONFIGURAÇÃO XML

```
<tc>
<filters>
<filter>
        <name>VOZ</name>
        <subfilter>
                <name>Megaco (H.248)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip sport 2945 0xff \
        match ip sport 2944 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>H.323</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 1300 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>SIP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 5060 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Pacotes VoIP usando RTP</name>
                <command>
tc filter add dev {interface} parent 1:0 protocol ip prio 0 handle 9 fw
flowid 1:{id}
iptables -t mangle -A FORWARD -p udp -m length --length 40:160 -j MARK --
set-mark 0x9
                </command>
        </subfilter>
</filter>
<filter>
        <name>Video</name>
        <subfilter>
                <name>RTSP (Real Time Sreaming Protocol)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 554 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Microsoft Media Services</name>
                <command>
```

```
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 1755 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Video usando Real Time Transport Protocol</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 5004 0xff \
        match ip sport 5005 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
</filter>
<filter>
        <name>Controle</name>
        <subfilter>
                <name>ECHO protocol</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 7 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>TACACS</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 49 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>DNS</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 53 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Finger</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 79 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>SQL Services</name>
```

```
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 118 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>NETBIOS</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 138 0xff \
        match ip sport 139 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>BGP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 179 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>LDAP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 389 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Who</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip sport 513 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>AODV</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 654 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>OLSR</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
```

```
        match ip sport 698 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Samba</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 901 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>NAS (Newest Admin System)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 991 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Emule (Nao-oficial)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip sport 4672 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Radmin</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 4899 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>UPnP (Windows)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 5000 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>NAT Port Mapping</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 5351 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
</filter>
<filter>
```

```
        <name>DADOS</name>
        <subfilter>
                <name>FTP (Dados e Controle)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 21 0xff \
        match ip sport 20 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>SSH</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 22 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Telnet</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 23 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>SMTP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 25 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>TFTP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip sport 69 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>HTTP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 80 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>POP3</name>
                <command>
```

```
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 110 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>SFTP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 115 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>IRC</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 194 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>HTTPS</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 443 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>FTP sobre SSL</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 989 0xff \
        match ip sport 990 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>IMAP sobre SSL</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 993 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>POP3 sobre SSL</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 995 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
```

```
        <subfilter>
                <name>OpenVPN</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip sport 1194 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Kazaa (Não-oficial)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip sport 1214 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>MySQL Data System</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 3306 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>VNC</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 989 0xff \
        match ip sport 5500 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>VNC (HTTP)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 989 0xff \
        match ip sport 5800 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Bit Torrent</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 6681 0xff \
        match ip sport 6682 0xff \
        match ip sport 6683 0xff \
        match ip sport 6684 0xff \
        match ip sport 6685 0xff \
        match ip sport 6686 0xff \
        match ip sport 6687 0xff \
        match ip sport 6688 0xff \
```

```
        match ip sport 6689 0xff \
        match ip sport 6690 0xff \
        match ip sport 6691 0xff \
        match ip sport 6692 0xff \
        match ip sport 6693 0xff \
        match ip sport 6694 0xff \
        match ip sport 6695 0xff \
        match ip sport 6696 0xff \
        match ip sport 6697 0xff \
        match ip sport 6698 0xff \
        match ip sport 6699 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
</filter>
<filter>
        <name>Mensageiros Eletronicos</name>
        <subfilter>
                <name>AOL Messenger</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip protocol 17 0xff\
        match ip sport 531 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Windows Live Messenger</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 1863 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Yahoo Messenger</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 5050 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>ICQ</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        match ip sport 6665 0xff \
        match ip sport 6666 0xff \
        match ip sport 6667 0xff \
        match ip sport 6668 0xff \
        match ip sport 6669 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
</filter>
<filter>
        <name>GERAL</name>
        <subfilter>
```

```
                <name>Todos os pacotes TCP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 06 0xff\
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>Todos os pacotes UDP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>ICMP</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 01 0xff\
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>IPv6</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 41 0xff\
        flowid 1:{id}
                </command>
        </subfilter>
</filter>
<filter>
        <name>Especiais</name>
        <subfilter>
                <name>Xbox Live (jogos online)</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} protocol ip u32\
        match ip protocol 17 0xff\
        match ip protocol 06 0xff\
        match ip sport 3074 0xff \
        flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>ACK</name>
                <command>
tc filter add dev {interface} parent 1:0 prio {prio} u32 \
     match ip protocol 6 0xff \
     match u8 0x10 0xff at nexthdr+13 \
     flowid 1:{id}
                </command>
        </subfilter>
        <subfilter>
                <name>ACK em pacotes menores que 64 bytes</name>
                <command>
tc filter add dev {interface} parent 1:0 protocol ip prio {prio} u32 \
            match ip protocol 6 0xff \
            match u8 0x05 0x0f at 0 \
            match u16 0x0000 0xffc0 at 2 \
            match u8 0x10 0xff at 33 \
            flowid 1:{id}
```

```
                </command>
        </subfilter>
</filter>
</filters>
<init>
#!/bin/sh

# Script criado pela ferramenta TC MESH em {data}

# Inicializacao dos modulos
INSMOD=/sbin/insmod;
grep -q ^sch_htb /proc/modules || $INSMOD /lib/modules/`uname -r`/sch_htb.o
grep -q ^sch_sfq /proc/modules || $INSMOD /lib/modules/`uname -r`/sch_sfq.o
grep -q ^cls_u32 /proc/modules || $INSMOD /lib/modules/`uname -r`/cls_u32.o
grep -q ^sch_dsmark /proc/modules || $INSMOD /lib/modules/`uname -
r`/sch_dsmark.o
grep -q ^sch_gred /proc/modules || $INSMOD /lib/modules/`uname -
r`/sch_gred.o
grep -q ^sch_prio /proc/modules || $INSMOD /lib/modules/`uname -
r`/sch_prio.o
grep -q ^sch_red /proc/modules || $INSMOD /lib/modules/`uname -r`/sch_red.o
grep -q ^sch_tbf /proc/modules || $INSMOD /lib/modules/`uname -r`/sch_tbf.o

# Limpando Regras
tc qdisc del dev {interface} root

</init>
<init-pc>
#!/bin/sh

# Script criado pela ferramenta TC MESH em {data}

# Limpando Regras
tc qdisc del dev {interface} root


</init-pc>
<qdisc>
        <htb>
# QDisc Pai
tc qdisc add dev {interface} root handle 1:0 htb default 0

# Classe Pai
tc class add dev {interface} parent 1:0 classid 1:1 htb rate {maxspeed}Mbit

# Subclasses e Filtros
        </htb>
        <tbf>
tc qdisc add dev {interface} root tbf rate {maxspeed}Mbit burst {burst}kb\
latency {latency}ms peakrate {peakrate}{peakrate_tax} minburst {minburst}
        </tbf>
        <sfq>
tc qdisc add dev {interface} root qdisc sfq perturb {perturb}
        </sfq>
        <dsmark>
tc qdisc add dev {interface} handle 1:0 root dsmark indices {indices}
default_index {default_index}

tc class change dev {interface} classid 1:1 dsmark mask 0x0 value 0x{tos}
        </dsmark>
        <prio>
tc qdisc add dev {interface} root handle 1: prio
```

```
        </prio>
</qdisc>

<subclass>
tc class add dev eth0 parent 1:{parent} classid 1:{id} htb rate
{htb_rate}{htb_tax} {ceil} {htb_max}{htb_max_tax}
</subclass>
<u32>
tc filter add dev {interface} protocol ip prio {prio} parent 1:0 u32 flowid
1:{id} u32 match ip {u32}
</u32>
<size>
tc filter add dev {interface} protocol ip prio {prio} parent 1:0 handle
{fwmark} fw flowid 1:{id}
iptables -t mangle -A FORWARD -m length --length {size_min}:{size_max} -j
MARK --set-mark 0x{fwmark}
</size>

</tc>
```

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo