View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositório Institucional do UniCEUB

Centro Universitário de Brasília – UniCEUB
Faculdade de Tecnologia e Ciências Sociais Aplicadas - FATECS
Curso de Engenharia da Computação

CLÁUDIO AMORIM DE SOUSA

Análise de Tráfego de Aplicações e Priorização de Pacotes em Redes TCP/IP

Brasília-DF
2009

CLÁUDIO AMORIM DE SOUSA

Análise de Tráfego de Aplicações e Priorização de Pacotes em Redes TCP/IP

Monografia apresentada como requisito parcial para a conclusão do curso de bacharelado em Engenharia da Computação do Centro Universitário de Brasília- UniCEUB

Orientador: Prof. MSc. Antonio José Gonçalves Pinto

Brasília-DF
2009

CLÁUDIO AMORIM DE SOUSA

Análise de Tráfego de Aplicações e Priorização de Pacotes em Redes TCP/IP

COMISSÃO EXAMINADORA

Prof. MSc. Antonio José Gonçalves Pinto
(Orientador)

Prof,
(Membro)

Prof.
(Membro)

Brasília-DF
2009

AGRADECIMENTOS

Agradeço primeiramente a Deus, aos colegas e amigos pelo apoio, e incentivo. Agradeço ao orientador Antonio José, pelo auxílio no desenvolvimento deste trabalho, aos meus pais, e irmãos por acreditarem na minha luta e perseverança durante todos esses anos.

DEDICATÓRIA

Dedico este trabalho à Nossa Senhora das Vitórias, a minha querida esposa e filha, Alanna Amorim e Maria Vitória, que foram minhas inspirações para superar todas as dificuldades encontradas no desenvolvimento deste trabalho ao longo deste semestre.

RESUMO

Empresas do setor de Tecnologia da Informação (TI) enfrentam bastantes dificuldades para garantir uma correta política de utilização dos recursos de rede através do enlace de dados. Este projeto tem por objetivo fornecer ao administrador de rede uma ferramenta completa, que proporcione a visualização gráfica do consumo em Kbps das aplicações que entram e saem da rede. Com base nas informações coletadas, é possível ter uma dimensão das aplicações que mais afetam o consumo da banda. O presente projeto fornecerá ao administrador aplicabilidade de qualidade de serviço (QoS) priorizando ou minimizando uma determinada parcela da banda para cada aplicação ou aplicações. Toda a interatividade de QoS é feita através de interface web.

Palavras chaves: HTB, SFQ, qualidade de serviço (QoS), Linux, tc, qdisc, rrdtool, sqlite, filas, Iptables, L7filter, bridge, DD-WRT, rrdupdate, rrdgraph, TCP/IP, Wan, Man, Thoughput.

ABSTRACT

Companies in the sector of Information Technology (IT) face many difficulties to ensure a correct policy for the use of network resources via the data link. This project aims to provide the network administrator a complete tool, which provides a graphic display of consumption in Kbps of the applications that enter and leave the network. Based on information collected, you can have a dimension most of the applications that affect the consumption of Data Link. This project will provide the administrator applicability of quality of service (QoS) prioritizing or minimizing a particular portion of bandwidth for each application or applications. All the interactivity of QoS is done through web interface.

Keywords: HTB, SFQ, qualidade de serviço (QoS), Linux, tc, qdisc, rrdtool, sqlite, filas, Iptables, L7filter, bridge, DD-WRT, rrdupdate, rrdgraph, TCP/IP, Wan, Man, Thoughput.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ADSL - Asymmetric Digital Subscriber Line. Sistema que permite a utilização das linhas telefônicas para transmissão de dados em velocidades maiores que as permitidas por um modem convencional

ATM – (Asynchronous Transfer Mode): protocolo de comunicação que permite o tráfego compartilhado de voz, dados e vídeo

Bittorrent – mecanismo utilizado para troca de arquivos em redes p2p

CLI – Command Line Interface

Delay - tempo de resposta do canal

DS – data source. Parâmetro utilizado no RRDTOOL

Firmware – Sistema operacional de um ativo de rede

Full-meshed – arquitetura de rede MPLS na qual todos os pontos de presença podem se interconectar entre si, sem a necessidade de um ponto de presença comum para roteamento

GNU - General Public License

IEEE – Institute of Electrical and Electronic Engineers

Iptables – ferramenta de filtro de pacotes IP utilizado em sistemas Linux

ISP - Internet Service Providers

Kbps – Kilo bits per second

Kernel - núcleo do sistema operacional que controla o hardware

Minix – Versão de sistema operacional utilizado como instrumento de ensino

MPLS - Multi-protocol Label Switching

OSI – Open Systems Interconnect

Overflow – estouro de pilha

P2P – peer-to-peer

QoS - quality of service

RFC – (Request for Comments): são padrões elaborados pelo órgão gerenciador da internet, o Internet Society.

RFC 821 – Simple Mail Transfer Protocol

RFC 822 - Standard for the format of ARPA Internet text messages

Roteador – equipamento tipicamente utilizado para fazer a interface entre uma rede local e uma rede de telecomunicações.

RRDTOOL – Round Robin Database Tool

RTP – Remote Time Protocol

SLA - service level agreement

SQLITE – sistema gerenciador de bando de dados SQL Lite

Switch – equipamento que comuta tráfego em uma rede local

TC – Traffic Control

TCP/IP – Transmition Control Protocol / Internet Protocol

Throughput - velocidade real de acesso atingida no instante presente do tráfego da aplicação

UDP – User Datagram Protocol

LISTA DE GRÁFICOS

CAPÍTULO 1. INTRODUÇÃO

1.1 Motivação

No início ano 2000, com o aumento da demanda no acesso à internet, criou-se um novo perfil de usuário corporativo que exigiu velocidades de acesso cada vez maiores, com disponibilidade de serviço vinte e quatro horas ao dia, sete dias na semana (24x7). Diante desse cenário, os provedores de serviço de internet (ISP – *Internet Service Providers)* tiveram que implementar novas tecnologias. São exemplos de tais tecnologias: MPLS (*Multi protocol Label Switching*), ATM E3 (34Mbps), IEEE 802.3ae (10 *Gigabit Ethernet*), ADSL2 (12 Mbps), etc.

Mesmo após implementação de novas tecnologias de acesso ao meio, que trabalham com velocidades comumente expressas em “Megas” (Mega bits por segundo - Mbps), não se atinge um padrão ideal de acesso, ou seja, não é alcançada uma correta política de utilização dos recursos distribuídos numa rede geograficamente distribuída.

Para uma análise criteriosa de utilização do uso da banda, visando um possível diagnóstico, o administrador de rede costuma ter acesso somente a informações restritas do provedor de serviço, que não detecta quanto e quando cada aplicação consumiu no enlace de dados.

A motivação deste projeto baseia-se na necessidade de se obter informações mais concisas, onde o administrador de rede não dependerá somente do provedor de serviço para extrair dados que geralmente são limitados e não auxiliam no diagnóstico de uma possível lentidão.

A ferramenta desenvolvida fornecerá ao administrador uma visão instantânea do fluxo de tráfego, através das informações coletadas e demonstradas graficamente. Em posse dessas informações pode-se otimizar os recursos de rede através da aplicação de QoS, via interface web, conseqüentemente melhorando a eficiência nos serviços de uma empresa.

1.2 Objetivo Geral do Trabalho

Numa rede distribuída, o administrador depara-se com situações momentâneas de extrema lentidão no acesso aos recursos de rede distribuídos, causando insatisfação nos usuários, que costumam reclamar de lentidões generalizadas.

Na maioria dos casos, a lentidão ocorre devido à super-utilização do enlace de dados, onde aplicações mais “pesadas” como HTTP, FTP, MS-SQL sobrepõem outras aplicações, ou mesmo causam lentidão no acesso a própria aplicação, como por exemplo, no download de um arquivo p2p via http, que simultaneamente compromete o acesso a navegação web.

O projeto tem um diferencial: Prover a identificação e a representação gráfica do tipo de aplicação a partir da camada de aplicação (modelo TCP/IP). Por mais que um software utilize a transferência de arquivos P2P via protocolo http, a ferramenta proposta neste trabalho irá diferenciar esse tráfego, dando subsídios para que o administrador de rede possa fazer sua identificação e

consequentemente gerenciar o tráfego que passa pela sua rede, definindo, a partir do QoS, prioridades separadas para cada aplicação, seja P2P, ou http (navegação web).

O projeto visa contribuir para implementação de uma correta política de utilização do enlace de dados, onde o administrador poderá gerenciar, através de interface web as configurações de Qualidade de Serviço (QoS), definindo a taxa de utilização para cada aplicação, por prioridade: Máxima, alta, média ou baixa.

1.3 Objetivos Específicos

A ferramenta desenvolvida neste projeto tem por objetivo proporcionar ao administrador de rede, uma visão gráfica das aplicações que trafegam em sua rede, com intuito de identificar a taxa de utilização (Kbps) do tráfego de entrada e saída, a fim de, dimensionar corretamente o QoS da rede, garantido determinada faixa de utilização para cada aplicação.

Para geração dos gráficos de aplicações é utilizado um método de registro de pacotes trafegados, onde são cadastradas no equipamento um total de 382 aplicações mais conhecidas em redes de computadores. Esta ferramenta possibilita:

- A visualização gráfica das oito aplicações que mais consomem banda, informando a respectiva taxa em Kbps de cada uma;
- A quantidade de tráfego (Bytes) gerada por aplicação;
- O throughput do link (Kbps);
- A aplicação de QoS para aplicações.

1.4 Justificativa e Relevância do Tema

Geralmente o que ocorre nas empresas, sejam públicas ou privadas, é a falta de criação de uma correta política de utilização dos recursos de rede. Mesmo quando existem políticas pré-definidas empregadas no ambiente institucional, elas são passiveis de serem burladas, o que pode comprometer o perfeito uso dos recursos de rede, ocasionando prejuízo a sua agilidade.

Os usuários, normalmente, não têm ciência da utilização correta dos recursos de rede, por exemplo, um simples “download” de um arquivo *bittorrent*, (uma aplicação *P2P*) na internet com tamanho de 64 MB, pode comprometer temporariamente e de forma generalizada o acesso as demais estações na rede, prejudicando a qualidade no acesso aos sistemas inerentes a empresa, o que causa prejuízos.

Dessa forma, com intuito de atender ao nível de qualidade tanto do atendimento, quanto do serviço prestado, seja de empresa privada ou orgão público, faz-se necessária a utilização do modelo da ferramenta utilizado neste projeto, que fornecerá ao administrador uma visão do que ocorre em sua rede e conseqüentemente terá subsídios para agir, através da aplicação de Qualidade de Serviço (QoS), priorizando os recursos de rede considerados essenciais (críticos).

1.5 Escopo do Trabalho

O projeto implementa uma ferramenta que irá registrar informações dos pacotes trafegados na rede e demonstrar graficamente:

- As oito aplicações que mais utilizam recursos da banda passante, informando a respectiva taxa em Kbps de cada uma;
- A quantidade de tráfego (Bytes) gerada por aplicação;
- O *throughput* do enlace de dados (Kbps).

A Qualidade de Serviço (QoS) para as aplicações utilizando policiamento pré-definido está inserido neste projeto, como forma de fornecer autonomia para que o administrador possa ajustar os recursos de rede conforme suas necessidades (prioridades).

1.6 Resultados Esperados

Uma correta política de utilização das aplicações distribuídas através de redes geograficamente distribuídas faz-se necessária a toda empresa em potencial, seja privada ou pública.

A visualização gráfica das 8 (oito) aplicações mais trafegadas na rede proposta neste projeto é um diferencial, pois são as aplicações que mais sobrecarregam o link. A aplicação de QoS poderá ser feita tendo como base as informações de tráfego dessas aplicações

O administrador de rede terá um equipamento que utiliza diversas ferramentas e tecnologias disponíveis no mercado. A partir da visualização gráfica de consumo das aplicações, o administrador poderá executar uma ação de QoS com interatividade direta, através de interface web. Espera-se ganhar agilidade, e a precisão no diagnóstico de uma possível lentidão nos recursos de rede podem ser comprovadas pelo administrador, através da análise gráfica disponível na ferramenta.

Todas as ferramentas utilizadas para o desenvolvimento deste projeto estão em acordo com a política de utilização e distribuição definida pelo GNU/GPLv3. [LOZANO, 2008].

1.7 Estrutura do trabalho

Este projeto de pesquisa está organizado em seis capítulos. Após a introdução feita no presente capítulo, o capítulo dois realiza a apresentação do problema pesquisado. No capítulo três é realizado o referencial teórico no qual o projeto está embasado. O capítulo quatro apresenta a proposta de solução e modelo que engloba a apresentação geral, o modelo de topologia e a descrição das etapas do modelo do projeto. O capítulo cinco contempla a aplicação da solução com resultados, onde é relatado o ambiente de simulação e implementação do modelo proposto, a descrição da aplicação da solução e a avaliação global do modelo de solução. Por fim, o capitulo seis apresenta a conclusão, onde se tem um resumo do que foi tratado no projeto, seus objetivos e se os mesmos foram atingidos com a

implementação do modelo em questão, e encerra com a sugestão de futuros trabalhos em acordo com o tema abordado neste projeto.

CAPÍTULO 2. APRESENTAÇÃO DO PROBLEMA

Com o aumento do volume de aplicações que trafegam na rede, provedores de serviço de internet (ISPs) se vêem obrigados a melhorar a infra-estrutura atual, assim como as tecnologias, a fim de prover velocidades cada vez mais rápidas no acesso a redes de áreas amplas (wan), haja vista a necessidade em integrar as diferentes demandas de recursos de rede, já que existe um leque muito extenso de aplicações com variadas necessidades, exigindo a classificação do tráfego e o estabelecimento de prioridades de encaminhamento.

Mesmo com a implementação de novas tecnologias, como a MPLS, ADSL2, o administrador de rede ainda se depara com problemas de lentidões demasiadas num curto intervalo de tempo, gerando insatisfação por parte dos usuários.

Para diagnosticar a causa das supostas lentidões na rede, os administradores, geralmente, não possuem ferramentas para análise de tráfego que chequem a taxa de utilização do enlace, ou seja, que possibilite fazer uma aferição do tráfego de rede. Quando o enlace atinge mais de 99,5% de utilização, é considerado saturado, e começa a perder pacotes (drop) no roteador, o que ocasiona um comprometimento no fluxo natural dos mesmos. [EMBRATEL, 2009]

Esse cenário vem se repetindo ao longo dos anos, onde o administrador de rede depara-se com situações em que tem dificuldade em visualizar o consumo da banda, assim como não tem informações de qual ou quais aplicações estão degradando o acesso, além de não possuir qualidade de serviço aplicado a rede.

A aplicação de qualidade de serviço na rede pode ser feita a partir do roteador, mas exige-se um profissional habilitado para configurá-lo através da linha de comando (CLI) específica daquele tipo de equipamento. Caso o parque computacional da empresa tenha diversos roteadores, com certa variedade de fabricante, torna-se um trabalho bastante árduo ao administrador, já que exigirá um domínio de todas as interfaces de linhas de comandos(CLIs) dos diversos fabricantes de roteadores utilizados pela empresa.

Este projeto implementa melhorias neste contexto, já que integra duas ferramentas em uma única, ou seja, disponibiliza a visualização gráfica do consumo de banda e do consumo das aplicações; e aplicabilidade de Qualidade de Serviço (QoS) à partir de interface web, onde se poderá selecionar as aplicações que serão submetidas ao mesmo, relacionando-as à partir de políticas pré-definidas, sem a necessidade de um conhecimento profundo sobre as diversas linhas de comando(CLI) para cada modelo de roteador disponibilizado no mercado .

CAPÍTULO 3. REFERENCIAL TEÓRICO E BASES METODOLÓGICAS

A principal tecnologia de rede utilizada neste projeto, a Fast Ethernet (IEEE 802.3u), utiliza como base o modelo de referência TCP/IP (Protocolo de Internet/ Protocolo de Controle de Transmissão).

Para tratamento e manipulação de pacotes, foi usado o Iptables, uma das ferramentas IP (protocolo de internet) mais respeitadas no âmbito da comunidade internacional. O Rrdtool, com sua flexibilidade matemática para tratamento dos dados, em especial em bits e bytes, a tornam ferramenta certa. Demais ferramentas utilizadas no projeto como tc para o qos, sqlite para banco de dados, dentre outras, são apresentadas no decorrer do capítulo.

3.1 O modelo de referência TCP/IP

O modelo TCP/IP – Transmission Control Protocol / Internet Protocol, foi desenvolvido pela rede de pesquisa ARPANET, ligada diretamente ao Departamento de Defesa Americano, originado da necessidade de interligar redes espalhadas geograficamente, que pudesse sobreviver a qualquer condição adversa, seja uma ameaça terrorista, política, ou nuclear.

A arquitetura TCP/IP é ilustrada na figura 01, sendo comparada com a arquitetura base, o modelo OSI.

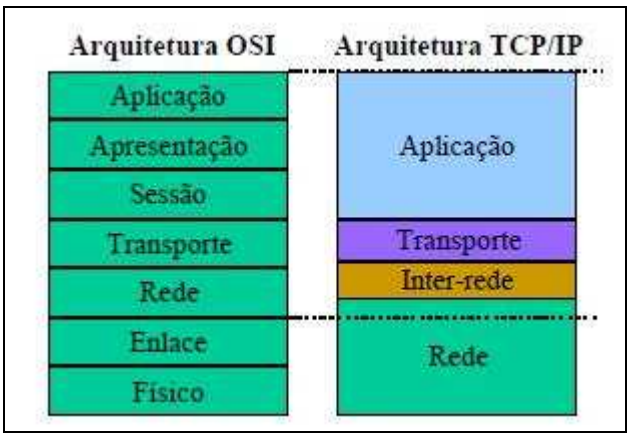

Fig. 01 - Comparativo entre a Arquitetura OSI e a Arquitetura TCP/IP
Fonte: BAETA (2001)

A seguinte ilustração detalha a Arquitetura TCP/IP, mostrando alguns dos principais protocolos para cada camada.

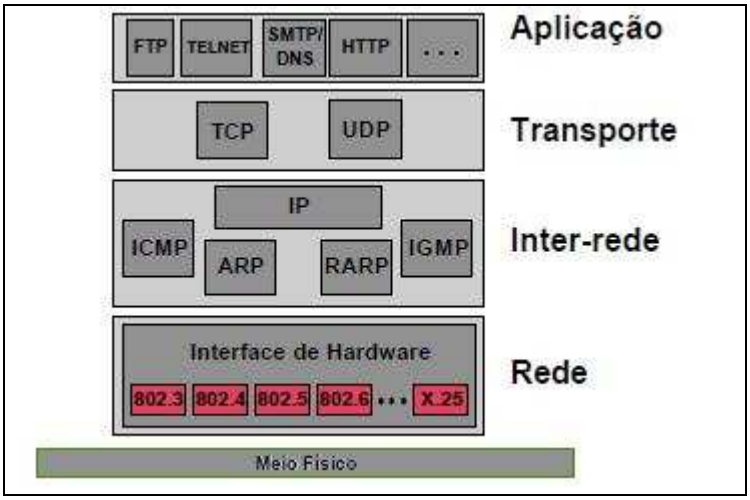

Fig. 02 – Protocolos da arquitetura TCP/IP
Fonte: BAETA (2001)

3.1.1 A Camada de Aplicação

O modelo TCP/IP não possui as camadas de sessão e apresentação. Como não foi percebida qualquer necessidade, elas não foram incluídas. A experiência com o modelo OSI provou a seguinte tese: elas são pouco usadas na maioria das aplicações. [TANENBAUM, 1997]

Tal camada é caracterizada por protocolos que solicitam à camada de transporte serviços orientados ou não orientados à conexão. A seguir, uma breve explicação sobre alguns dos principais protocolos dessa camada: [BAETA, 2001]

- HTTP: *Hyper Text Transfer Protocol* – É considerado como um dos motivos para a grande popularização da Internet. Tal protocolo permite a transferência de arquivos entre computadores em uma rede baseada em TCP/IP através de textos com uma formatação especial que permite a utilização de recursos gráficos avançados e navegação com uso de hyperlinks.

- SMTP: *Simple Mail Transfer Protocol* – Protocolo padronizado pelas RFC´s 821 e 822 para a troca de mensagens de correio eletrônico entre computadores em redes TCP/IP. Ao enviar determinada mensagem, o SMTP inclui na mesma uma informação de controle indicando o caminho pelo qual a mesma foi enviada.

- SNMP: *Simple Network Management Protocol* – É um protocolo de gerenciamento usado vastamente em redes TCP/IP. Tal protocolo permite o acesso a informações gerenciais de componentes descritos como nós gerenciados. Tais informações podem ser enviadas como *traps* ou lidas manualmente pelas estações de gerenciamento.

- FTP: *File Transfer Protocol* - Protocolo de transferência de arquivos que pode operar tanto em modo texto quanto em modo binário. Tal protocolo utiliza-se do protocolo de transporte TCP, o que torna a conexão mais confiável e com mecanismos de recuperação de erros.

- DNS: *Domain Name Server* - Tem por função principal promover a resolução de nomes de hosts (p.ex. www.trf1.gov.br) para endereços ips que serão utilizados, de fato, para o endereçamento das informações.

- TELNET: Protocolo de aplicação mais popular utilizado para emulação de terminais através de redes TCP/IP.

- TFTP: *Trivial File Transfer Protocol* - Protocolo similar ao FTP, porém muito mais simplificado, pois não tem suporte a conexões confiáveis, ou seja, utiliza o transporte UDP. Tal protocolo é comumente utilizado em equipamentos de redes locais (ativos de rede) para que se possa modificar arquivos de configuração dos mesmos.

- RTP: *Real Time Protocol* – Definido pela RFC 1889, tal protocolo foi projetado para fornecer suporte a aplicações que necessitam de serviços sensíveis ao tempo, tais como conexões de voz e videoconferência.

3.1.2 A camada de transporte

A camada 3 na arquitetura TCP/IP (não confundir com a camada 3 no modelo OSI) é a camada de transporte. Os PDU´s referentes a tal camada recebem o nome de Pacotes. Ou seja, o popular termo "pacotes de redes" só se aplica a PDU´s (package datagram unit) da camada de transporte. [BAETA, 2001]

Os dois principais protocolos de tal camada são o TCP (Transmission Control Protocol) e o UDP (User Datagram Protocol).

- TCP: Caracterizado por oferecer um serviço confiável entre aplicações. Para tanto, antes que os dados sejam transferidos, é aberta uma conexão entre as parte comunicantes através de um mecanismo conhecido como *handshake*. Ao término da transferência, ocorrerá a liberação da conexão (*release)*. Além de o protocolo ser orientado a conexões, existem alguns outros mecanismos, tal como controle de erros nos cabeçalhos (checksum) e controle do sequenciamento dos pacotes.

- UDP: Protocolo de transporte que oferece um serviço não-confiável para aplicações. No caso do UDP, não existe orientação à conexão e nem mecanismos de correção de dados. Tal protocolo é tipicamente utilizado por aplicações sensíveis ao tempo, tais como conexões de voz, ou pelas outras que não necessitam de entrega confiável de dados. Uma das principais funções da camada de transporte é a multiplexação. Ou seja, tal camada é capaz de associar o mesmo endereço de rede (camada de inter-rede) a diversos processos de usuário que são executados sobre a camada de aplicação. Tal conceito é feito através de **portas** que identificam serviços específicos. Por exemplo: a porta 23 é utilizada pelo protocolo de aplicação TELNET. A porta 53 é utilizada pelo serviço de nomes DNS. Dessa forma, um mesmo host de determinada rede pode ser um servidor DNS e um servidor TELNET com o mesmo endereço de rede, dependendo-se das portas TCP ou UDP associadas às requisições.

3.1.3 A camada Inter-Rede

A camada de Inter-rede corresponde ao segundo nível da arquitetura TCP/IP. Os principais protocolos alocados em tal camada são o ICMP (Internet Control Message Protocol), IGMP (Internet Group Management Protocol), o ARP (Address Resolution Protocol), o RARP (Reverse Address Resolution Protocol) e o famoso protocolo IP (Internet Protocol).

O protocolo ICMP tem por objetivo promover mensagens de controle na comunicação entre nós num ambiente de rede TCP/IP. Dessa forma, tal protocolo permite que tais nós tomem conhecimento sobre o estado de outros nós. Tais mensagens podem informar, por exemplo, que determinada máquina não está ativa, ou que não existe rota para um nó de destino, bem como outras mensagens de controle.

O protocolo ARP tem a função de resolver/associar endereços IP´s, (IPV4) para endereços MAC (Media Access Control), geralmente constuídos pela base hexadecimal, atrelados fisicamente na NIC (Network Interface Card).

O protocolo RARP (Reverse Address Resolution Protocol) atua de forma oposta ao ARP. Ou seja, tal protocolo é usado na resolução de endereços MAC em endereços IP. Tal serviço é útil, em particular, para a implementação do serviço conhecido como DHCP (Dynamic Host Configuration Protocol). Tal serviço tem a finalidade de associar ou de fornecer dinamicamente endereços IP disponíveis em uma faixa configurável para as máquinas que assim solicitarem. Tal solicitação é feito através do envio de uma requisição RARP, que envia o endereço físico do adaptador para o endereço de broadcast, solicitando o endereço IP para o mesmo.

O Protocolo IP é responsável pela comunicação entre máquinas em uma estrutura de rede TCP/IP. Tal comunicação é feita por um serviço sem conexão e não-confiável. Assim, qualquer tipo de serviço que deva ser orientado a conexão ou confiável deve ser fornecido pelos protocolos de níveis superiores. As funções mais importantes realizadas pelo protocolo IP é o endereçamento, que é independente do endereçamento físico das camadas inferiores e da própria topologia da rede utilizada, e o roteamento entre redes IP distintas.

3.1.4 A camada de Rede ou Host/Rede

Abaixo da camada de inter-rede, encontra-se a camada de rede, ou host/rede, onde geralmente esta camada não é descrita ou informada explicitamente em livros, exceto pela descrição resumida que o host deve-se conectar com a rede utilizando um protocolo, que proporcione o envio de pacotes IP. [TANENBAUM, 1997]

Este projeto utiliza como base o modelo de arquitetura TCP/IP. Como é a principal arquitetura difundida mundialmente, tornou-se quase que obrigatória a utilização desta arquitetura pelos fabricantes mundiais, tanto em nível de hardware quando em nível de software. Neste projeto, as ferramentas que o compõem, utilizam como base este modelo.

No projeto, a identificação dos pacotes tanto para construção dos gráficos, quanto para aplicação de QoS, é feita através das camadas 3 (transporte) e 4 (aplicação).

3.2 O Linux

Para Rubem E. Ferreira (2003), o Linux é um clone de Unix criado como uma alternativa barata e funcional para quem não está disposto a pagar o alto preço de um sistema Unix comercial ou não tem um computador suficiente rápido.

Em 1983, Richard Stallman fundou a Free Software Foundation (Fundação de Software Livre), cujo projeto, GNU, tinha por finalidade criar um clone melhorado e livre do sistema operacional Unix, mas que nao utilizasse seu código-fonte.

O desafio do GNU era enorme. Havia a necessidade de desenvolver o kernel (núcleo do sistema operacional que controla o hardware), utilitários de programação, de administração do sistema, de rede, comandos-padrão, etc. Porém, no final da década de 1980, o projeto tinha fracassado: apenas os utilitários de programação e os comandos padrão estavam prontos, e o kernel, não.

Nessa mesma época, vários esforços independentes para desenvolver clones do Unix estavam em andamento. O Dr. Andrew Tanenbaum desenvolveu o Minix como instrumento de ensino, baseando-se no microprocessador Intel 8086, por estar amplamente disponível e barato. O Minix era útil no ensino dos princípios estruturais dos sistemas operacionais.

Entretanto, o 8086 não dispunha de memória virtual ou de memória protegida e só endereçava 1 MB de cada vez. Isso era uma enorme barreira para um sistema operacional moderno, multitarefa. Por isso, desde o seu início, o Minix estava limitado a ser um instrumento de ensino.

Linus Benedict Torvalds era aluno da Universidade de Helsinque, na Finlândia, no final da década de 1980. Ele percebeu que o Intel 80386 era o único microprocessador disponível na época capaz de executar um clone do Unix. Além disso, ainda que o 80386 não fosse barato, ele era o único disponível. A sua opção por este microprocessador foi uma escolha correta, pois garantiu a ele, posteriormente, o grande número de voluntários que tornou o desenvolvimento do Linux viável até hoje.

Linux B Torvalds, estava disposto a construir um kernel clone do Unix que possuísse memória virtual, multitarefa preemptiva e capacidade de multiusuários. Era um trabalho gigantesco e, na prática, impossível para apenas uma pessoa concluí-lo, ainda que estivesse familiarizada com as complexidades dos sistemas operacionais.

Na primavera de 1991, Linus B. T. iniciou seu projeto particular, inspirado no seu interesse pelo Minix. Ele limitou-se a criar, em suas próprias palavras, “um Minix melhor que o Minix”. E depois de algum tempo de trabalho em seu projeto solitário, conseguiu criar um kernel capaz de executar os utilitários de programação e os comandos-padrão do Unix clonados pelo projeto GNU.

Em 5 de outubro de 1991, Linux Torvalds lançou a primeira versão “oficial” do Linux: o Linux 0.02. A partir dessa data, muitos programadores no mundo inteiro tem colaborado e ajudado a fazer do Linux o sistema operacional que é atualmente.

No desenvolvimento deste projeto, a utilização do Linux como sistema base, foi fundamental, pois todas as ferramentas utilizadas neste projeto foram desenvolvidas para a arquitetura Linux.

3.3 O Iptables

O filtro de pacotes do kernel do Linux 2.4 funciona por meio de regras estabelecidas na inicialização do sistema operacional. Todos os pacotes entram no kernel para serem analisados. As chains (correntes) são as situações possíveis dentro do kernel. Quando um pacote entra no Linux, o kernel verifica o destino dele e decide qual chain manipulará esse pacote. Isto é chamado do roteamento interno. Os tipos de chains irão depender da tabela que está sendo utilizada no momento. Existem três tabelas possíveis. O programa Iptables fornece uma interface para que o usuário possa manipular o filtro de pacotes do kernel. [FERREIRA, 2003]

O subsistema de processamento de pacote de rede kernel do Linux é denominado Netfilter, e o Iptables é o comando utilizado para sua configuração.

A arquitetura Iptables agrupa regras de processamento de pacotes de rede dentro de tabelas por função (filtragem de pacotes, tradução de endereços de rede e outros pacotes “mangling” – “alterados”), cada um dos recursos tem chains (sequências) de regras de processamento. As regras consistem em “matches”, (utilizados para determinar a quais pacotes a regra será aplicada) e alvos (que determinam o que será feito aos pacotes coincidentes). [PURDY, 2005]

Netfilter é uma parte muito importante do kernel do Linux em termos de segurança, pacotes e manipulação. O front end para o netfilter é o iptables.

As características do iptables são filtrar pacotes e proporcionar o Nat (tradução de endereços de rede). Existem várias outras coisas que podem ser feitas com o iptables, como manipulação de pacotes usando filtro *Layer* 7.

Na figura 03, é apresentada a passagem do pacote através das **tabelas** e de suas **chains** (sequências):

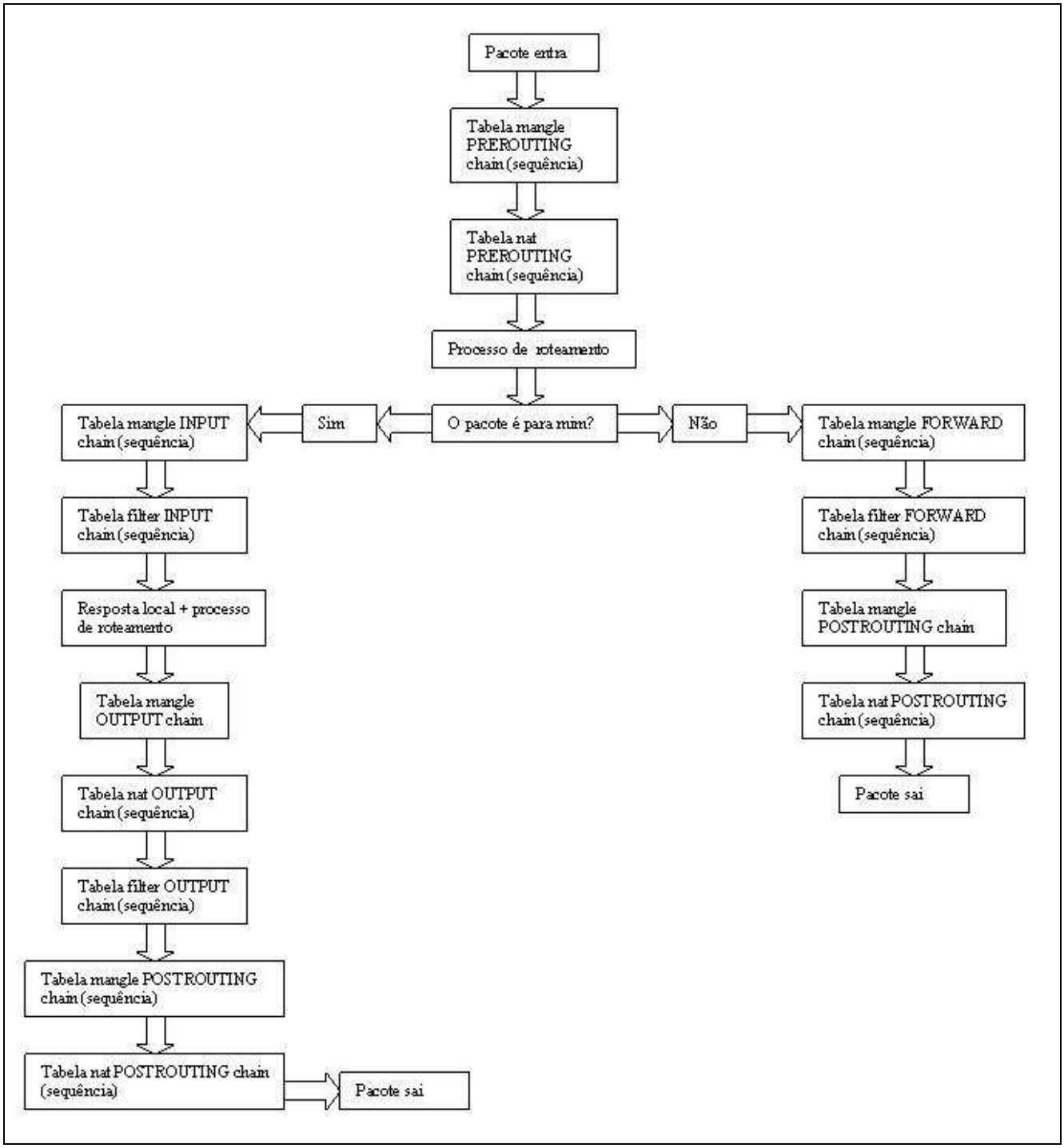

Fig. 03 - Diagrama de operação do Iptables, utilizando tabelas filter, mangle, nat

O iptables utiliza três tipos de tabelas, que são utilizadas para definir quais tipos de chains serão usadas.

- **filter**: é a tabela padrão, sendo usada quando nenhuma tabela for especificada. È usada quando há tráfego normal de dados, sem a ocorrência de NAT (Tradução de Endereços de Rede). Usa as chains INPUT, OUTPUT, FORWARD.

- **nat**: é utilizada quando há NAT. Exemplo: passagem de dados de uma rede privada para a internet. Usa as chains PREROUTING, POSTROUTING, OUTPUT.

- **mangle**: é utilizada para efetuar alterações especiais em pacotes. Usa as chains PREROUTING e OUTPUT. [FERREIRA, 2003]

O iptables contém algumas tabelas, onde cada uma contém um conjunto de regras padrão, que são as chains:

- **INPUT**: contém regras para pacotes destinados a máquina Linux.
- **FORWARD**: contém regras para pacotes que a máquina Linux faz o roteamento para outro endereço IP.

- **OUTPUT**: contém regras para pacotes que são gerados a partir da máquina Linux. [GHEORGHE, 2006]

- **PREROUTING**: Analisa todos os pacotes que estão entrando no firewall para sofrerem NAT. O PREROUTING pode fazer ações de NAT com o endereço de destino do pacote. Isso é chamado de DNAT (Destination NAT).

- **POSTROUTING**: Analisa todos os pacotes que estão saindo do firewall para sofrerem NAT. O POSTROUTING pode realizar ações de NAT com o endereço de origem do pacote. Isso é chamado de SNAT (Source NAT). [FERREIRA, 2003]

Para registro dos pacotes com intuito de construir os gráficos das aplicações, as informações de contadores de pacotes e bytes do Iptables são fundamentais para essa funcionalidade, pois seus contadores são coletados a cada minuto pela ferramenta, através das tabelas e chains manipuladas para o projeto (ver scripts iptables.sh) atualizando os gráficos conforme os contadores do Iptables são carregados à medida que os pacotes passam pela bridge.

A ferramenta mais indicada tanto para registro das aplicações, quanto para marcação dos pacotes no QoS, é o Iptables.

O módulo netfilter com o iptables vem compilado no kernel do Linux desde a versão 2.4.

3.3.1 Iptables - identificando pacotes através da camada de aplicação (Layer 7)

Para que os gráficos sejam gerados, o Iptables deve contabilizar a quantidade de pacotes trafegadas para uma determinada aplicação. A maneira que é feita a contabilização está diretamente relacionada à porta (TCP/UDP) que a aplicação utiliza.

Dessa forma, como a porta está associada à aplicação, todas as conexões destinadas a esta porta são contabilizadas. O problema ocorre quando há aplicações que não utilizam portas dinâmicas, como por exemplo, aplicações FTP, RTP, Bittorrent, H323 etc, tornando inviável sua identificação. Para resolver esse problema à comunidade internacional livre, criou em 2003 o projeto L7 Filter protocol.

3.4 O projeto L7 Filter Protocol

O desenvolvimento do L7 filter começou em 2003 em resposta a aplicações proprietárias que já faziam a checagem de pacotes em nível de aplicação, também conhecido como “packet shapping” (método de controle de grande parcela da banda, usada por protocolos específicos, também conhecidos como "arbitração da banda" e “qualidade de serviço”) usadas por empresas proprietárias. [LEVANDOSKI, 2009]

O intuito era proporcionar uma ferramenta de código aberto que classificasse os pacotes em nível layer 7 (ref. a camada de aplicação modelo OSI) abrindo a competitividade neste mercado.

Em maio de 2003, foi liberado a primeira versão do l7-filter, um pacote que adicionava um filtro nos sistemas de QoS do kernels do Linux.

Em outubro de 2003, os desenvolvedores decidiram que não era tão bom a interação do projeto com um framework de QoS, então foi decidido liberar uma versão para o Netfilter. A versão 1.0 foi lançada em janeiro de 2005.

Devido ao l7-filter ter sido implementado usando o netfilter, ele é capaz de classificar tudo através de matches, usando, por exemplo, o Iptables.

Pensando em alguma forma de captura para as aplicações que possuem portas dinâmicas, foi desenvolvido o projeto L7-filter protocol. que possui peculiaridade na identificação da aplicação através da leitura do pacote IP.

O modulo L7-filter é composto por diversas assinaturas, que são strings (padrões) usadas no cabeçalho IP para caracterizar o tipo de aplicação. Dessa forma, para o projeto a escolha dessa ferramenta trás um diferencial, pois é possível identificar diversas aplicações que usam portas dinâmicas como exemplo: RTP, H.323, FTP, bitrorrent, Edonkey, counterstrike, smb, etc. Dessa forma, além de conseguir identificar as aplicações em nível l7 para construção dos gráficos, é possível aplicar QoS fazendo um match (identificação) a partir do Iptables para aplicações l7.

Isso torna o projeto robusto, pois a identificação da aplicação é precisa tanto para a geração dos gráficos quanto para aplicação de QoS.

3.5 A ferramenta RRDTOOL

A ferramenta de código aberto, RRDTOOL – Round Robin Database Tool, foi criada em março de 1999 desenvolvida por Tobias Oetiker.

Como sua própria sigla descreve, o RRDTOOL usa uma técnica conhecida como, Round Robin Databases (RRDs), que trabalha com um montante de dados fixos, interagindo funções matemáticas para construção de diversos tipos de gráficos, a serem definidos pelo projetista.

3.5.1 Tipos de dados funcionais com o RRDTOOL

Números inteiros, decimais, octais, como exemplo, os contadores *ifInOctets* de um roteador são dados que podem ser coletados. Outro exemplo seria dados para monitoramento da temperatura do condicionamento de ar de uma unidade do centro de processamento de dados (CPD) de uma empresa.

Ambos os dados podem ser coletados via SNMP (Simple Network Management Protocol - RFC 4789 - protocolo difundido pela sociedade da internet - Internet Society), onde o administrador pode trazer as informações coletadas de vários dispositivos para que o RRDTOOL possa gerar graficamente de diversas formas essas informações. [OETIKER, 2009]

3.5.2 O rrdcreate

Para criar um arquivo com base na base de dados RRD (Round Robin Database), os tipos de medição dos dados para geração dos gráficos, são definidos por parâmetros disponíveis no código da ferramenta, existindo 5 tipos de DS (Data Source), ou origem de dados, para medição. São eles:

- GAUGE: usado, por exemplo, na medição de temperatura, ou um determinado número de pessoas numa sala, ou o valor de um compartilhamento Red Hat.

- COUNTER: usada para medição de incremento dos dados, como as informações dos contadores de octetos (*ifInOctets)* num roteador. A origem de dados (DS) uilizando o **COUNTER**, assume que o contador nunca decrementa, com exceção quando um contador atinge o *"overflow".*

- DERIVE: característica de armazenar a derivação de uma linha indo do último para o valor atual do dado de origem. Esta opção pode ser usada, por exemplo, para medir a taxa de pessoas entrando e deixando uma sala.

- ABSOLUTE: usado para contadores que reiniciam após terem sido lidos. É usado para contadores rápidos que tendem a estourar (overfow). A intenção é reiniciar este DS depois de ter sido lido, para dar o máximo de intervalo disponível antes do próximo overflow. Um outro uso deste DS, é para coisas que pode-se contar como número de mensagens desde a última atualização.

- COMPUTE: Tem a função de armazenar o resultado de uma fórmula aplicada a outros DS no RRD.

Para o projeto, o tipo de dado de origem (DS) usado foi o **COUNTER**, devido à peculiaridade dos tipos de dados coletados, pacotes e bytes, que são dados incrementais.

A syntax utilizada no rrdcreate, obedece:

rrdtool create filename [--start|-b start time] [--step|-s step] [DS:ds-name:DST:dst arguments] [RRA:CF:cf arguments]

- Filename: O nome do arquivo rrd que será criado. Arquivos rrd devem terminar com a extensão .rrd.
- --start|-b start time (default: now - 10s): Especifica o tempo em segundos desde 1970/01/01 UTC, quando o primeiro valor deve ser acrescentado ao RRD. O Rrdtool não aceitará quaisquer dados antes desta data especificada.
- --step|-s step (default: 300 seconds): Especifica o intervalo em segundos, que os dados (valores) serão alimentados no rrd.

3.5.3 O rrdupdate

A função rrdupdate alimenta novos valores em um arquivo RRD. Os valores são interpolados de acordo com as propriedades do arquivo .rrd para a qual os dados são gravados.

A atualização dos valores é feita através de comando personalizado pelo administrador, como exemplo, usado neste projeto: *http=`$iptables -L FORWARD -v -x |awk '/spt:www/{print $2}'`,* que tem a função de pegar o valor de bytes da aplicação www.

Os valores também podem ser atualizados através de banco de dados. O principal parâmetro que o rrdupdate utiliza é: *N | timestamp: valor [: valor ...]*

O timestamp defini o período de tempo, onde cada valor é atualizado. Este tempo pode ser definido em segundos desde 1970/01/01 ou usando a letra "N", caso em que o tempo de atualização está definido para ser o tempo atual. Valores de tempo negativo são subtraídos do tempo atual.

Aproveitar o momento certo para o segundo é especialmente importante quando você está trabalhando com dados de fontes do tipo COUNTER, DERIVE ou ABSOLUTE.

O formato dos valores adquiridos a partir da fonte de dados depende do tipo de fonte de dados escolhida. No projeto utiliza-se tanto valores numéricos de variáveis declaradas para o caso do throughput da rede, quando também valores numéricos capturados a partir do banco de dados.

Normalmente os valores são numéricos, mas a aquisição de dados em módulo podem impor a sua própria análise deste parâmetro, enquanto os dois pontos (:) continua a ser o separador da fonte dos dados. Neste exemplo:

rrdtool update demo1.rrd N: 3.44:3.15: U: 23

Significa: atualize o arquivo de banco de dados demo1.rrd com 3 conhecidos e um valor desconhecido.

Use o tempo atual como a atualização de tempo.

3.5.4 O rrdgraph

A comando rrdgraph é utilizado para geração dos gráficos, sendo que, é necessário alimentar (atualizar) os dados em determinados períodos, tarefa que é feita pelo rrdupdate. O rrdgraph necessariamente não está limitado a um banco de dados.

O rrdgraph funciona perfeitamente a partir de dados gerados, por exemplo, por variáveis declaradas.

Às vezes, os dados visualizados graficamente não são exatamente no formato que deveria ser. Por exemplo, pode-se coletar bytes por segundo, mas deve-se mostrar bits por segundo.

O comando para fazer o cálculo em bits por segundo deve ser definido através da syntax disponível no comando rrdgraph, que seria um subcomando, através de expressões matemáticas definidas pelo RPN (rrdgraph_rpn). Exemplo: *CDEF: mydatabits = Mydata, 8, **

A flexibilidade da escolha do tipo de dado de origem (DS) GAUGE, COUNTER, DERIVE a ser usada no tratamento dos valores pelo RRDTOOL torna esta ferramenta a mais indicada para o desenvolvimento do projeto, pois há escolha de tratamento dos dados.

No projeto foi escolhido DS (data source) **COUNTER**, devido à peculiaridade dos tipos de dados coletados, pacotes e bytes, que são dados incrementais.

A utilização do comando RRDUPDATE torna-se vital, pois têm o objetivo de alimentar novos valores do contador gerado pelo Iptables. Por exemplo, à medida que o tráfego FTP aumenta, a quantidade de bytes trafegados nessa aplicação é lida pelo RRDUPDATE e consequentemente enviada ao RRDGRAPH que tem a tarefa da geração gráfica mostrando a nova taxa, e para isso precisa-se dos valores atualizados pelo RRDUPDATE a cada minuto.

Para visualização de informações como exemplo em Kbps no eixo y do gráfico, deve-se fazer cálculos especiais, por exemplo, na conversão de bytes em bits, utiliza-se no rrdgraph a expressão CDEF, exemplo:

"CDEF:trafego_in_20min_bits=trafego_in_20min,8,"* \

Onde, *trafego_in_20min* é o DS (data source) cadastrado no arquivo, exemplo rtime-20min.rrd, e o 8 é o multiplicador usado para o cálculo e visualização gráfica da velocidade em Kbps, gerado no arquivo rtime-20min.png.

Pode-se usar um banco de dados em conjunto com o RRD para enviar e recolher os valores numéricos para geração gráfica, a fim de automatizar o processo.

3.6 O SQLITE

O SQLITE é um banco de dados, baseado no SGBD (Sistema Gerenciador de Banco de Dados) SQL, com recursos básicos, como o próprio nome descreve “LITE”, que proporciona os parâmetros necessários a manipulação de dados em tabelas. [HWACI, 2009]. O código fonte do SQLITE está num domínio público.

Os responsáveis pelo desenvolvimento e manutenção do SQLITE são formados pelos patrocinadores: Mozilla; empresa desenvolvedora do browser Firefox; Adobe; principal desenvolvedora de aplicativos sociais, Bloomberg. líder mundial em tecnologia de informações financeiras; Symbian; líder de mercado de sistema operacional aberto para dados avançados em smartphones.

O SQLITE é diferente do mecanismo usado em outros bancos de dados SQL, pois é bem simples de ser usado, sendo sua principal característica seu tamanho e sua velocidade de gravação e entrega dos dados, além da operação confiável.

Para garantir o seu objetivo, o SQLITE teve que sacrificar outras características que algumas pessoas acham usuais, como: concorrência alta, alto índice de funções de construção, procedimentos de armazenamento, XML e/ou extensões Java, escalabilidade tera ou peta-byte, dentre outras.

A regra básica para saber quando usar o SQLITE inclui: Use o SLITE para situações simples de administração, implementação, e manutenção consideradas as diretivas mais importantes, comparadas as características complexas dos mecanismos de bancos de dados empresariais.

3.6.1 Áreas de atuação do SQLITE

O Sqlite opera em diversas áreas, desde dispositivos menores, a aplicações consolidadas, para sítios da internet. Esta seção descreve algumas áreas de atuação.

3.6.1.1 Dispositivos embutidos e aplicações

A base de dados SQLite exige pouca ou quase nenhuma administração. O SQLite é uma boa escolha para dispositivos e serviços que devam funcionar sem vigilância humana e sem apoio.

O SQLite é ideal para uso em telefones celulares, PDAs, set-top boxes, e / ou aparelhos. Também funciona como um banco de dados embutido para download de aplicações. [HWACI, 2009]

3.6.1.2 Websites

SQLite geralmente pode trabalhar com uma grande engine de base de dados para tráfego de websites de médio e baixo porte. A quantidade de tráfego na Web que o SQLite pode lidar, depende naturalmente da forma como o site utiliza (nível de intensidade) sua base de dados. [HWACI, 2009]

3.6.1.3 Banco de dados temporário ou uso interno

Para programas que tenham um número muito grande de dados que devam ser ordenados em diversas formas, muitas vezes é mais fácil e rápido carregar os dados em uma base de dados *SQLite* e utilizar consultas através de *selects* (seleções) *ORDER BY* (ordenando por) para extrair os dados na forma e da ordem necessárias ao invés de tentar executar o mesmo código repetidas vezes em operações manuais. [HWACI, 2009]

No decorrer do desenvolvimento do projeto, principalmente na busca das aplicações mais utilizadas em redes, definidas por dois perfis de acesso, grupos web e grupos interativos, os scripts de busca estavam demorando a retornar com os valores, devido ao shell script estar pesquisando 17 ocorrências, uma de cada vez para um total de 382 aplicações.

Dessa forma, para garantir dinamismo na busca das aplicações foi utilizado o banco de dados *SQLITE*, sendo o banco de dados mais indicado para o projeto, pois atendeu a necessidade de inserção e captura de valores, depositando e trazendo as informações requisitadas de forma muita rápida e eficiente. A implementação do banco aliviou consideravelmente o processamento de hardware do D-link / DIR-320.

3.7 O firmware DD-WRT

O projeto DD-WRT foi fundado em dezembro de 2004. O alvo inicial foi fazer uma modificação no firmware original dos equipamentos da Linksys, em especial roteadores wireless, proporcionando suporte simples na autenticação Radius. O desenvolvedor decidiu adicionar mais algumas ferramentas “leves” ao projeto, apenas por divertimento. [GOTTSCHALL, 2009]

Em janeiro de 2005, depois de o projeto ter sido disponibilizado publicamente na internet, houve por parte da comunidade internacional uma grande repercussão, o autor decidiu desenvolver mais e mais ferramentas e começou também a redefinir completamente o firmware.

Em maio de 2006 o projetista fez a fusão com a NewMedia-NET, lançando oficialmente o firmware DD-WRT.

Em setembro de 2006, a versão DD-WRT V23 SP2 foi liberada com grande sucesso. Neste momento também foi decidido desenvolver o firmware para várias plataformas incluindo: Ávila Gateworks, X86, FON Fonera, dentre outras. Segundo o autor, por alto, o projeto foi adequado para 90 tipos de roteadores diferentes.

Em agosto de 2007, finalmente sua equipe conseguiu um contrato com o principal vendedor de roteadores wireless, a Buffalo. A Buffalo decide que querem trabalhar juntos, em equipe, com os responsáveis do firmware DD-WRT, para projetos de futuras plataformas wireless.

Em outubro de 2007, o equipamento Buffalo WHR-G54DD (Estados Unidos apenas) foi lançado pela Buffalo com um exclusivo firmware DD-WRT.

Em maio de 2008, foi lançada a versão free v24, com ativação livre para todos os dispositivos de perfil residencial, incluindo os roteadores com chipset broadcom.

A empresa DD-WRT é formada por: Sebastian Gottschall (fundador, e desenvolvedor principal), Christian Scheele (Chris, CEO), Peter Steinhäuser (CEO), Ankush Malhotra (Maksat), Ales Majdic (Eko, Developer), Sylvain Bothorel (Botho, router webdesign), Felix Fietkau (nbd/openwrt, madwifi), Elke Scheele (Online Shop), Markus Quint (suporte), e um grande número de pessoas que prestam suporte ao redor do mundo.

A escolha desta ferramenta foi vital para o projeto, pois no DD-WRT roda o sistema base Linux na versão de kernel 2.4. O desafio deste projeto foi torná-lo executável num hardware limitado, composto por 32 MB de RAM, 4 MB de flash, processador de 240MHz MIPS32® (chipset broadcom BCM 5354), 16KB de instruções cache.

Como o hardware possui uma porta USB, originalmente criada pela D-Link para ser usada como servidor de impressão (Print Server), a empresa DD-WRT, fez proveito dessa porta, desenvolvendo um firmware, onde foi possível conectar um pendrive, montando um sistema de arquivos (file system) usando ext2. Dessa forma, todo o projeto está armazenado e montando num sistema de arquivos ext2, em um pendrive de 1GB, sendo executado diretamente no DIR-320.

3.8 O gerenciador de pacotes Optware

Com a última versão do DD-WRT (v24 SP1), é possível instalar os pacotes do optware. Os pacotes optware estão adaptados para o DD-WRT expandindo características completas do Linux no roteador, através de armazenamento *storage* (jffs, SD/MMC Card, USB) todos disponíveis nos pacotes optware. Esta característica permite ponto de montagem a partir de um pendrive usando o sistema de montagem /jffs. [OLEO, 2009]

O *jffs* é um sistema de arquivos estruturado utilizado em memória flash. Este ponto de montagem foi utilizado no projeto, pois o pendrive utiliza memória *flash* e o equipamento, DLink – DIR320, tem suporte através da porta *USB*.

No projeto, através do gerenciador de pacotes Opware, foram instalados os seguintes pacotes:

- thttpd-php (Servidor web php);
- Bash;
- Tc;
- bridge-utils;
- rrdtool;
- sqlite;

3.9 O pacote Iproute2 e a ferramenta Traffic Control (TC)

O iproute2 é o pacote que proporciona várias ferramentas para roteamento avançado, túneis e controle de tráfego. O iproute2 foi originalmente desenvolvido por Alexey Kuznetsov, e atualmente seu mantenedor é Stephen Hemminger. [GHEORGHE, 2006]

3.9.1 O Traffic Control (Controle de Tráfego)

A ferramenta *tc* (Traffic Control) permite aos administradores de rede, construir diferentes policiamentos de QoS, usando o sistema operacional Linux.

Os policiamentos de QoS no tc são conhecidas como queuing disciplines (disciplinas de filas) que possuem dois tipos: classless e classful.

Para aplicar QoS em uma máquina Linux, usa-se o comando tc que possui três variações:

- tc qdisc - manipula as disciplinas de fila;
- tc class - manipula classes;
- tc filter - manipula filtros usados para identificar pacotes.

No anexo 1, têm-se o descritivo da syntax utilizada pelo comando tc.

3.9.2 As disciplinas de Fila de Classes (Classless qdiscs)

Classless qdisc é uma disciplina de fila que não possui configuração para subdivisão interna.

São simples porque apenas permitem (accept), (drop), (delay) ou remarcam um pacote. Podem ser atribuídas a uma interface para fazer *shapping*.

Existem uma série de filas implementadas no Linux, a maioria delas incluídas no kernel do Linux:

- **FIFO (pfifo and bfifo)**: A mais simples qdisc, que opera no modo de regra First In, First Out. O algoritmo FIFO tem um limite de tamanho de fila (buffer size), que pode ser definido nos pacotes para *pfifo* ou em bytes for *bfifo*.

- **Pfifo_fast**: A qdisc padrão em todas as interfaces Linux. Essa fila é uma fila do tipo FIFO (First In, First Out), o que significa que nenhum pacote recebe tratamento especial. Essa fila é subdividida em três “bandas”. Em cada uma delas, a regra FIFO se aplica. Contudo, se houver pacotes aguardando para serem processados na banda 0, a banda 1 não será processada. O mesmo acontece para a banda 2, em relação a banda 1.
  O kernel respeita a flag TOS (Type Of Service) [RFC1349, 1992] do campo TCP/IP, e insere pacotes com “mínimo atraso” (minimum delay) na banda 0.

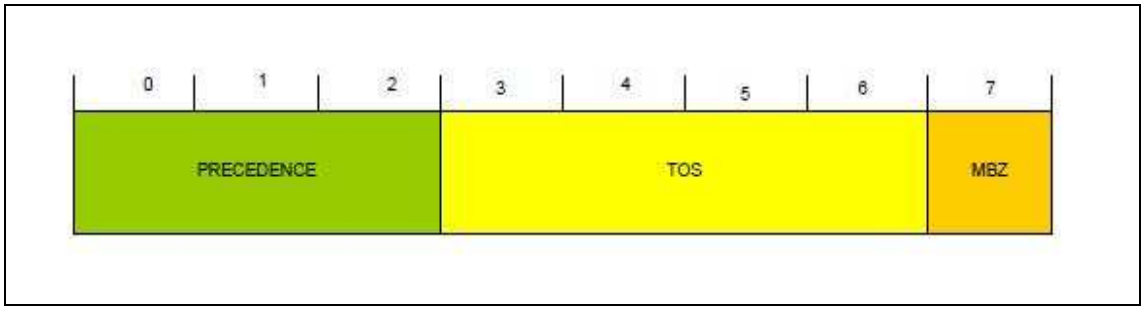

Fig. 04 - Octeto TOS do pacote [RFC1349, 1992]

Fonte: [MARTINS, 2005]

- **Stochastic Fair Queue (SFQ):** uma das mais amplas qdisc usadas. SFQ proporciona bastante distribuição de transmissão de dados sobre uma quantidade demasiada de fluxo.
- **Enhanced Stochastic Fair Queue (ESFQ):**
- **Random Early Detection and Generic Random Early Detection (RED and GRED):** qdisc usada para backbone, para taxas acima de 100 Mbps.

No projeto a qdisc utilizada é a SFQ, devido à peculiaridade no tratamento das filas para quantidade demasiada de fluxo de tráfego, ou seja, se o link estiver totalmente cheio e se deseja ter certeza de que nenhuma sessão simples domine a banda de saída, o ideal é utilizar o SFQ.

3.9.3 Característifcas do Stochastic Fairness Queueing (SFQ)

SFQ é uma implementação simples do algoritmo da família “fair queueing”. É menos apurado do que outros, mas também requer menos cálculos, enquanto permanece quase que perfeitamente justo. [MARTINS, 2005].

A palavra-chave na implementação do SFQ é conversação (ou fluxo), o que corresponde a uma sessão TCP ou um stream UDP. O tráfego é dividido em um grande número de filas FIFO, uma para cada conversação.

O tráfego é então enviado no modo “round-robin”, permitindo a cada sessão que envie dados. Isso leva a um comportamento extremamente justo e proíbe que uma simples conversação tome conta do link e interfira em outras conversações.

O SFQ é também chamado de estocástico porque ele na verdade não aloca uma fila para cada sessão, ele tem um algoritmo que divide o tráfego em um número limitado de filas usando um algoritmo de hash.

Por causa do hash, múltiplas sessões podem terminar no mesmo bucket, o que irá dividir cada chance de cada sessão enviar um pacote, dividindo a velocidade efetiva disponível. Para impedir que essa situação se torne perceptível ao usuário final, a fila SFQ troca seus algoritmos de hash com freqüência para que quaisquer duas sessões que colidam só o façam por um pequeno espaço de tempo (alguns segundos).

Existem alguns parâmetros a serem configurados no SFQ:

- Perturb: Reconfigura o hashing na quantidade de segundos definido por esse parâmetro. Se não for definido, o hash não será reconfigurado nunca.

- Quantum : Quantidade de bytes que um stream é permitido desenfileirar antes que a próxima fila seja processada. O padrão é 1 maximum sized packet (MTU-sized). Não se deve definir um valor menor que o MTU.

- Limit: O número total de pacotes que podem ser enfileirados por essa fila SFQ (antes desta começar a descartálos).

3.9.4 As disciplinas de filas de classes (classfull)

Uma disciplina de fila de classe (ou classful) é uma disciplina de fila que contém múltiplas classes. Esta qdisc é usada para diferenciar vários tipos de dados. A disciplina de fila classful mais comum é o CBQ (Class Based Queuing) e o HTB (Hierarchical Token Bucket). No projeto foi usada a qdisc HTB, pois é qdisc ideal para cenários menos complexos. Portanto foi empregada neste projeto a qdisc HTB.

A Figura 05 ajuda a entender as disciplinas de filas:

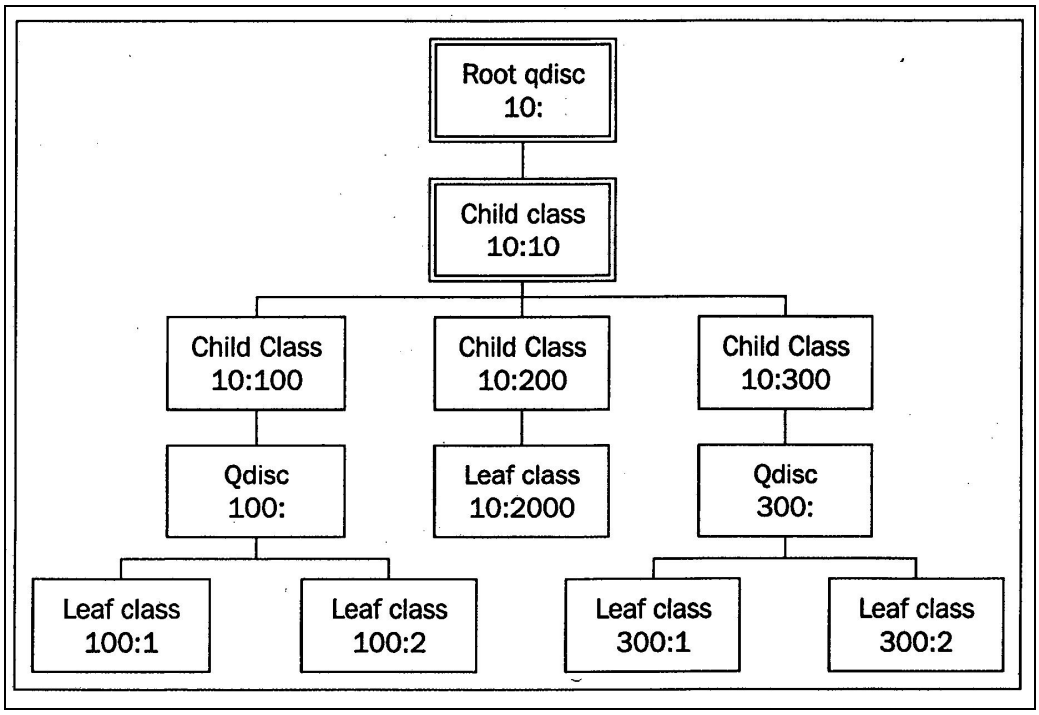

Fig. 05 - Disciplinas de filas de classes empregadas no CBQ e HTB

As disciplinas de fila classfull operam em forma hierárquica. Primeiramente cada interface tem uma qdisc (principal) root, que fala com o kernel. Segundo, existe uma classe filha (child class) anexada

a qdisc root. Para cada child class, pode-se anexar qdisc posteriores, ou leaf classes que são child classes da qdisc anexada a primeira child class, ou pode-se criar leaf class como child class anexas a qdisc root. Dessa forma, existem várias sequência para atender a necessidades de políticas de QoS a serem empregadas pelo administrador.

CAPÍTULO 4 – PROPOSTA DE SOLUÇÃO E MODELO

Neste capítulo, são apresentadas as etapas de desenvolvimento do projeto. Desde a descrição do modelo utilizado (topologia), passando por informações do esquemático (fluxograma), das ferramentas utilizadas, métodos empregados, das aplicações de regras, configurações, etc.

4.1 Apresentação Geral do Modelo Proposto

A figura 06 apresenta o modelo de Fluxograma utilizado neste projeto. O pacote IP, passa pela bridge, logo atravessa o filtro Iptables, depois é feito o registro de tráfego das aplicações, utilizado para construção dos gráficos. O mesmo pacote pode ou não ser marcado pelo qos, dependendo da definição do administrador. Caso o qos seja confirmado, é pacote passa a ser marcado e manipulado pela tabela mangle - Iptables para priorização no tráfego.

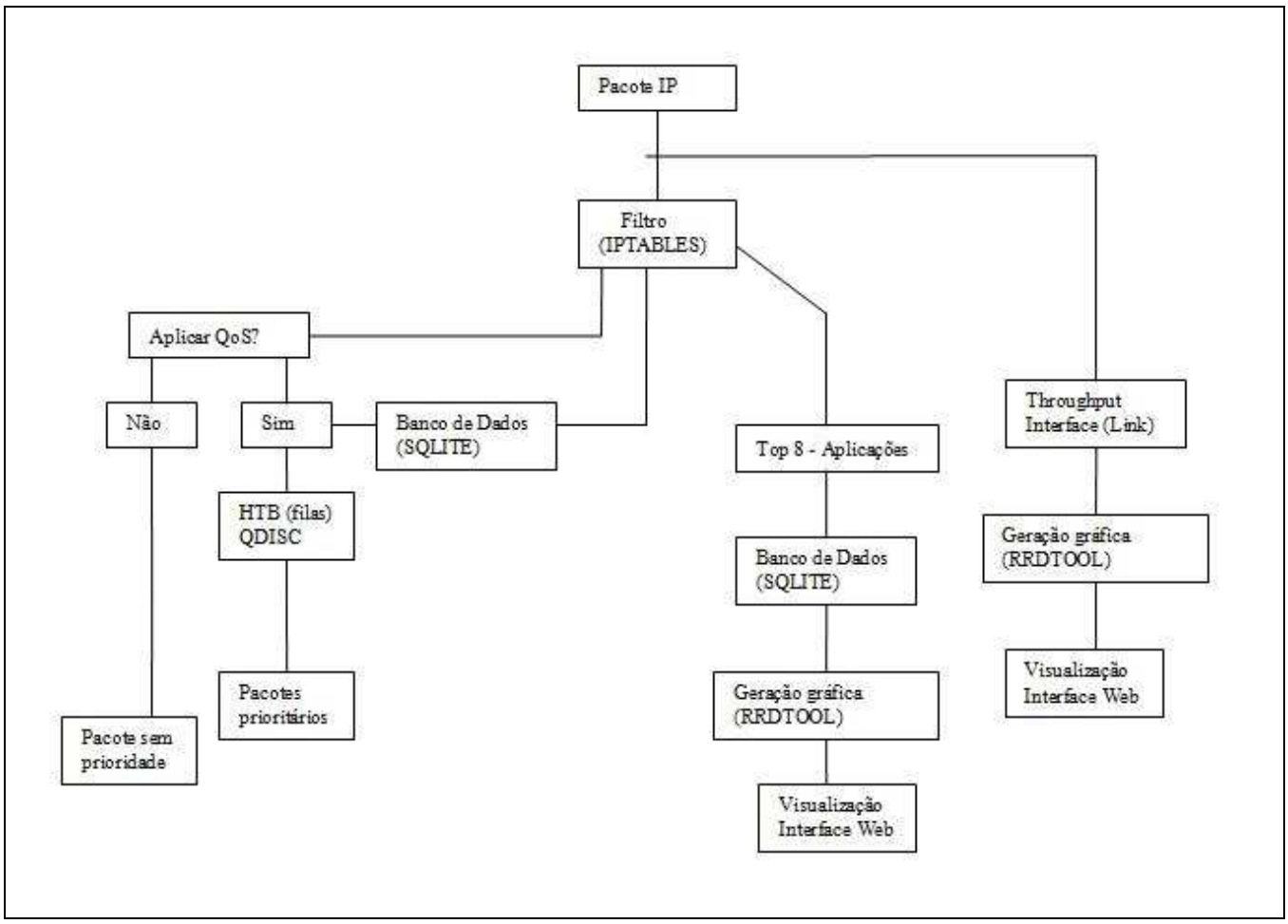

Fig. 06 - modelo de fluxograma utilizado no projeto

A Figura 07 mostra uma das topologias utilizada neste projeto. Esta topologia será aplicada na apresentação do projeto para a banca examinadora.

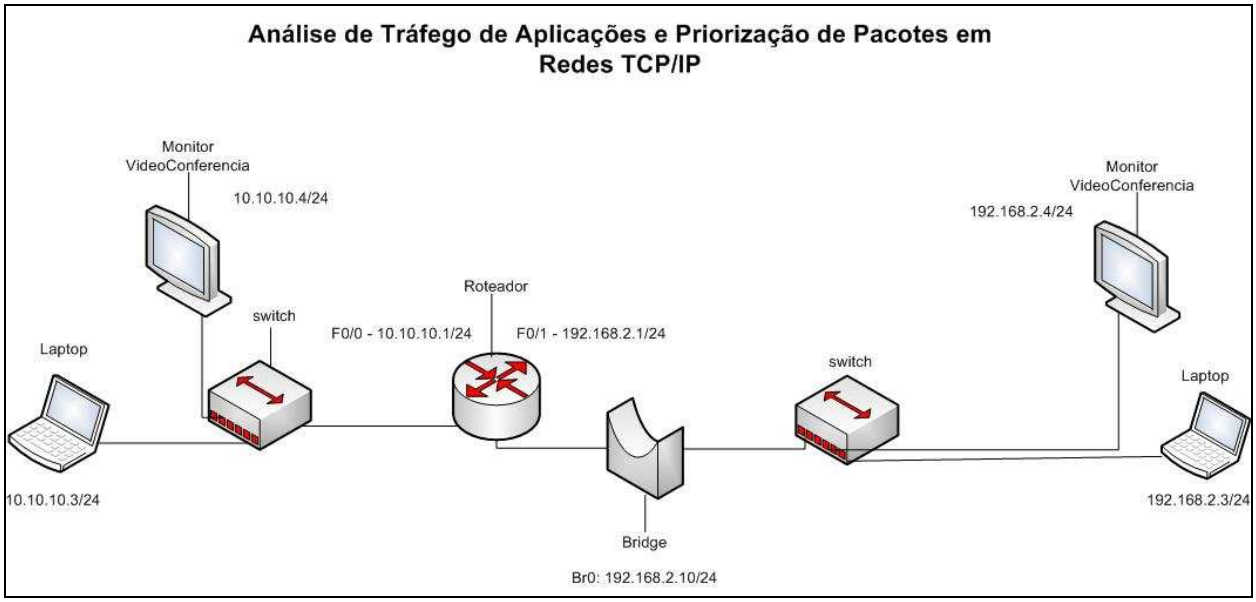

Fig. 07 – Modelo de Topologia empregada no projeto

Neste esquemático o roteador é o ativo responsável pelo roteamento dos pacotes entre as subredes. Existem dois switchs de borda, um em cada subrede, sendo estes, comutadores de pacotes responsáveis pela distribuição dos pacotes destinados aos hosts de rede.

A ferramenta desenvolvida faz o papel da bridge, sendo o equipamento responsável pelo registro do tráfego entrante e sainte da subrede 192.168.2.0/24. A bridge contabiliza a quantidade de pacotes, assim como define as aplicações que serão manipuladas para o qos, sendo as mesmas, priorizadas ou minimizadas, a livre escolha do administador.

A bridge deve ser instalada entre o roteador e o switch da rede. Essa posição foi definida estrategicamente, pois todo o pacote que tiver destino e origem a subrede 10.10.10.0/24, passará pela bridge.

Exemplificando essa etapa: A requisição de acesso a uma videoconferência, ou a um donwload de um arquivo é feita pelos hosts da subrede 192.168.2.0/24 destinado a subrede 10.10.10.0/24. Os pacotes trafegados em ambas as direções são contabilizados pelo Iptables.

A partir de um intervalo de tempo é possível ter uma visão gráfica do consumo de utilização das aplicações.

Foi simulado um enlace de dados metropolitano com banda de 1024 Kbps. Todos os pacotes passarão pela bridge, sendo feito o registro dos pacotes trafegados tanto na entrada quanto na saída.

Dessa forma, torna-se precisa a medição gráfica das aplicações trafegadas na rede, assim como o throughput real da mesma, pois todos os pacotes destinados ou originadas das subredes 192 e 10 passarão pela bridge.

A aplicação de QoS é feita na bridge, onde o administrador de rede seleciona quais aplicações serão priorizadas.

4.2 Descrição das Etapas do Modelo

Nesta seção estão descritas as etapas de desenvolvimento do projeto, desde a configuração do hardware utilizado a configurações de QoS.

4.2.1 O hardware utilizado

O hardware utilizado neste projeto é da marca D-Link modelo DIR-320, constituído de um processador MIPS32® de 240MHz, utilizando 32 MB de RAM. A limitação deste equipamento, conforme descrita, é bastante restrita, sendo que, no desenvolvimento do projeto foi necessário adicionar um série de controles para que os processos fossem executados por meio de filas, evitando assim a sobrecarga no processamento.

O sistema base constituído no DIR-320 é um firmware proprietário da D-Link®. Para o projeto, foi feita a formatação do firmware original do equipamento, sendo posto em seu lugar o firmware Linux DD-WRT na versão 24-SP2.

Utilizando o principal recurso que o equipamento oferece, um porta USB, foi possível montar um sistema de arquivo a partir do pendrive de 1GB, usando filesystem ext2.

A descrição da instação do firmware DD-WRT e a configuração do sistema de arquivos a partir do pendrive, estão descritas nas próximas duas seções.

4.2.2 Instalação do Firmware DD-WRT no D-Link DIR-320

Para instalar o Firmware DD-WRT no roteador D-Link DIR-320, é necessário fazer o download do firmware *dd-wrt.v24_usb_generic.bin* dispnonível em versão gratuita através do site do DD-WRT.

Para fazer o upload do firmware para o dir-320 é necessário ter um cliente de TFTP, que também está disponível em versão gratuita no próprio site do DD-WRT:

Tendo o firmware e o cliente TFTP, pode-se iniciar com a segunda etapa da instalação. Um cabo de rede deve ser disponibilizado para conexão entre a placa de rede do computador a porta número 1 do DIR. Na configuração TCP/IP da placa de rede, deve ser definido o ip estático 192.168.0.10.

Executando o cliente TFTP, deve-se definir o ip do servidor 192.168.0.1 (DIR), e o caminho do firmware baixado, cujo arquivo deve ser renomeado para firmware.bin.

Na terceira etapa, deve-se ligar o DIR, e após 3 segundos clicar no botão “upgrade” no cliente TFTP para iniciar o processo de upload de firmware. O Firmware original da D-Link será apagado, logo após será copiado o firmware DD-WRT.

Após a confirmação da instalação do firmware, pode-se reiniciar o DIR. No mesmo instante deve-se alterar o ip estático da placa de rede para: 192.168.1.2. O acesso via interface web pode ser feito através do ip: 192.168.1.1. Um usuário e uma senha serão solicitados no primeiro acesso, basta definir.

Dessa forma, a instalação do firmware DD-WRT no D-Link DIR-320 foi concluída.

4.2.3 Montando sistema de arquivos (File System) a partir do pendrive

Originalmente a porta USB disponível no DIR, foi feita para servidor de impressão (printserver), que proporciona o compartilhamento de uma impressora em um rede. Os desenvolvedores do firmware DD-WRT viram um proveito bastante útil para essa porta: montar um sistema de arquivos a partir dela, usando um pendrive ou HD externo.

Para fazer o mapeamento do sistema de arquivos no DIR, é necessário editar algumas configurações na interface web. Na aba “services”, deve-se habilitar as funcionalidades:

- Core USB Support;
- USB 2.0 Support;
- USB Printer Support;
- USB Storage Support, ext2 / ext3 File System Support;
- Automatic Drive Mount;
- Run-on-mount Script Name “/jffs/opt/etc/rc.local”;
- Disk Mount Point “/jffs”.

Após esse passo, deverá ser iniciado o processo de formatação do pendrive para sistema de arquivos ext2. Antes é preciso habilitar o serviço sshd localizado em: Services, Services, Secure Shell. O sshd foi habilitado para permitir acesso ao DIR via ssh, porta 22. Para formatar o pendrive deve-se logar no equipamento e executar o comando mkfs.ext2 /dev/discs/disc0/disc. Obs: confirmar o caminho da montagem do USB no campo “Disk Info” (Services – USB). Após a formatação digitar reboot.

Deve-se logar novamente no DIR, e digitar as seguintes linhas de comando para montagem da partição jffs e o swap automaticamente no reboot:

mkdir /jfss/opt
mkdir /jffs/opt/etc
vi /jffs/opt/etc/rc.local

Adicionar no arquivo rc.local, as seguintes linhas:

#!/bin/sh

/bin/mount -o bind /jffs/opt /opt
/opt/bin/busybox swapon /jffs/myswap.swp

Para atribuir permissão de execução no arquivo rc.local:
chmod +x /jffs/opt/etc/rc.local

Para habilitar o gerenciador de pacotes do dd-wrt deve-se digitar as linhas:

mount –o bind /jffs/opt /opt
wget http://www.wlan-sat.com/boleo/optware/optware-install-ddwrt.sh -O - | tr -d '\r' > /tmp/optware-install.sh
sh /tmp/optware-install.sh

Por fim, para habilitar o swap no dd-wrt:

/opt/bin/ipkg-opt update
/opt/bin/ipkg-opt install busybox
dd if=/dev/zero of=/jffs/myswap.swp bs=1k count=64000
/opt/bin/busybox mkswap /jffs/myswap.swp
/opt/bin/busybox swapon /jffs/myswap.swp

Obs: count igual 128000 cria uma area de swap (file) de 64 Mega bytes.

4.2.4 Configurando a Interface Bridge - br0

Para permitir o fluxo de pacotes transparente entre o roteador e o switch core da rede, é necessário que a ferramenta opere em modo bridge.

O DIR, possui 4 (quatro) interfaces Lan, e 1 (uma) Wan, descrita no equipamento como “Internet”. O firmware DD-WRT já possui o pacote da bridge (**bridge-utils**) instalado.

Dessa forma, foram definidas as interfaces que funcionariam em modo bridge, são elas: vlan1 e vlan0.

Arquivo de configuração ***regras*** (configuração modo bridge):

```
...
/bin/echo CONFIGURANDO A BRIDGE...

#adicionando a interface vlan1, vlan0 -> br0
/usr/sbin/brctl addif br0 vlan1
/usr/sbin/brctl addif br0 vlan0

#definindo ip na br0
#ifconfig br0 192.168.1.1 netmask 255.255.255.0 - ipdefault do DD-WRT
/sbin/ifconfig br0:3 192.168.2.10 netmask 255.255.252.0

#definindo rota
route add -net 0/0 gw 192.168.2.1 dev br0
...
```

O arquivo de regras está disponível no Anexo 1.

4.2.5 Geração Gráfica do Throughput do Link Wan

Para geração gráfica do throughput do link, foram utilizadas as seguintes ferramentas: rrdcreate, rrdupdate, rrdgraph, ifconfig, thttp:

Todo tráfego de pacotes que passa pela bridge, é registrado no kernel do Linux, onde cada interface de rede possui estatísticas da quantidade de pacotes trafegados, tanto dos pacotes recebidos (RX), quanto dos pacotes enviados (TX), através do comando *ifconfig*.

Tirando proveito desta característica, foi desenvolvido script que coleta informações através da interface vlan1 (interface wan), buscando os valores numéricos disponíveis nos parâmetros “RX bytes” e “TX bytes”.

Esses valores são carregados pelos comandos:

Para o tráfego de entrada:

```
…
bit=8
valor_in=`$ifconfig vlan1 |awk -F: '/RX.bytes/{print $2--}'`
trafego_in=`expr $valor_in \* $bit`
…
```

Para o tráfego de saída:

```
...
valor_out=`$ifconfig vlan1 |awk -F: '/TX.bytes/{print $3--}'`
trafego_out=`expr $valor_out \* $bit`
…
```

Tendo esses valores atualizados a cada minuto, o rrdupdate, alimenta o DS (data source) criado no rrdcreate, arquivo “*rtime-20min.rrd*” tanto do tráfego de entrada quanto de saída, através do comando:

```
...
rrd=/opt/bin/rrdtool
path=/opt/share/www/rtime/rtime-20min.rrd
$rrd update $path N:$trafego_in:$trafego_out
…
```

Já com os valores atualizados no arquivo *rtime-20min.rrd*, o rrdgraph pode iniciar a geração gráfica do throughput do Link dos últimos 20 minutos definidos através do script (rtime-at-20min.sh):

Faz-se observação nos comandos abaixo, que convertem o valor lido do rtime-20min.rrd de **bytes** para **bits**, usando a syntax de cálculo CDEF que tem exatamente a função de fazer cálculos a partir dos valores do DEF (Definition File).

```
"CDEF:trafego_in_20min_bits=trafego_in_20min,8,*"  \
"CDEF:trafego_out_20min_bits=trafego_out_20min,8,*"  \
```

```
#--------------gerando o RRDGRAPH
rrd=/opt/bin/rrdtool
path=/opt/share/www/rtime

$rrd graph $path/rtime-20min.png -a PNG -t "Throughput Link" --vertical-label 'bits por segundo' \
DEF:trafego_in_20min=$path/rtime-20min.rrd:trafego_in_20min:AVERAGE \
DEF:trafego_out_20min=$path/rtime-20min.rrd:trafego_out_20min:AVERAGE \
VDEF:"trafego_in_20min"max=trafego_in_20min,MAXIMUM \
VDEF:"trafego_in_20min"avg=trafego_in_20min,AVERAGE \
VDEF:"trafego_out_20min"max=trafego_out_20min,MAXIMUM \
VDEF:"trafego_out_20min"avg=trafego_out_20min,AVERAGE \
--start end-20min \
--zoom 1.2 \
"CDEF:trafego_in_20min_bits=trafego_in_20min,8,*"  \
"CDEF:trafego_out_20min_bits=trafego_out_20min,8,*"  \
AREA:trafego_in_20min#00FF00:Entrada \
COMMENT:"Maxima " \
GPRINT:"trafego_in_20min"max:"%6.2lf %Sbps" \
COMMENT:"Media " \
GPRINT:"trafego_in_20min"avg:"%6.2lf %Sbps\l" \
LINE2:trafego_out_20min#0000FF:Saida \
COMMENT:"Maxima " \
GPRINT:"trafego_out_20min"max:"%6.2lf %Sbps" \
COMMENT:"Media " \
GPRINT:"trafego_out_20min"avg:"%6.2lf %Sbps\l"
```

4.2.6 Geração Gráfica das oito aplicações mais trafegadas na rede

Para geração das oito aplicações mais trafegadas na rede, foram utilizadas as seguintes ferramentas: sqlite, iptables, rrdcreate, rrdupdate, rrdgraph, html.

4.2.6.1 Criação do banco de aplicações no kernel

O primeiro passo consiste na criação do banco de aplicações. O DD-WRT trás na sua versão de firmware 2.4, o pacote L7filter compilado.

As assinaturas L7, estão disponíveis no caminho: /etc/l7-protocols/protocols/. Somando-se com as portas mais conhecidas e utilizadas pelas redes corporativas (arquivo /etc/services), existem neste projeto um banco com 382 aplicações, ou seja, 112 aplicações definidas no arquivo /etc/l7-protocols/protocols/ e 276 definidas no arquivo /etc/services.

O banco de aplicações é criado através do script *iptables2.sh*. O script é executado somente uma vez, após o "*boot"* do equipamento.

A seguir é apresentado a sequência de comandos do arquivo executável iptables2.sh:

1. *#!/opt/bin/bash*
2. *iptables -F*
3. *iptables -X*
4. *touch /tmp/regras*
5. *#cria regras para os protocolos L7*
6. *for a in /etc/l7-protocols/protocols/**
7. *do*
8. *x=`echo $a|sed 's/.*cols.//;s/.pat//;'`*
9. *iptables -A FORWARD -i br0 -m layer7 --l7proto $x -j ACCEPT*
10. *done*

11. *#cria regras para os protocolos conhecidos*
12. *grep "tcp\|udp" /etc/services | while read a b c*
13. *do*
14. *x=`echo $b |sed "s/\//:/"|awk -F: '{print "iptables -A FORWARD -i br0 -p " $2 " --sport " $1 " -j ACCEPT" }'`*
15. *y=`echo $b |sed "s/\//:/"|awk -F: '{print "iptables -A FORWARD -o br0 -p " $2 " --dport " $1 " -j ACCEPT" }'`*
16. *#echo $x*
17. *$x*
18. *#echo $y*
19. *$y*
20. *done*

21. *#cria regra para os protocolos desconhecidos*
22. *iptables -A FORWARD -j ACCEPT*

23. *rm -f /tmp/regras*

Na linha 2, "iptables –F", limpa todas as tabelas do Iptables. A linha 3 "iptables –X" limpa as chains.

Na sequência o comando "touch /tmp/regras" é um dos controles de execução feitos no projeto, que significa: enquanto estiver executando este script, crie o arquivo /tmp/regras, após o término apague-o "rm -f /tmp/regras" (última linha).

Da linha 6 a 10, é executado um loop *for* que faz a varredura das 112 aplicações, nível L7 (l7filter). Logo após, através do comando *do*, é executado na linha 8 o comando: "*x=`echo $a|sed 's/.*cols.//;s/.pat//;'`"* , onde é utilizado o comando de edição de texto sed, ou seja, a partir da expressão /etc/l7-protocols/protocols/100bao.pat , substitua a expressão até " *.cols.// " por nada, onde aparecerá

100bao.pat, aim.pat, aimwebcontent.pat, etc. Depois, no mesmo sed “ ;s/.pat//;`` ” substitua tudo que for “ .pat ”, por nada “ // ”. Dessa forma, será atribuído na variável x todos os protocolos na sequência:

100bao
aimwebcontent
applejuice
ares
armagetron
battlefield1942
...

Na linha seguinte (9) é executado o comando “ *iptables -A FORWARD -i br0 -m layer7 --l7proto $x -j ACCEPT* ” , que irá construir o banco de aplicações l7 no kernel do Linux.

Na linha 12, é executado o comando: “ *grep "tcp\|udp" /etc/services* | *while read a b c ",* ou seja, mostre através do grep, a relação de linhas que possuem a ocorrência “tcp” e “udp” , do arquivo /etc/services. Depois na mesma linha “pipe”, “ while read a b c ” .

O while read, vai associar o nome da aplicação (a), o número da porta e tipo de protocol, tcp ou udp (b), e o nome da aplicação (c).

Na linha 13, é executado mais um “ *do* ” para construção do banco de aplicações mais utilizadas em redes, através do arquivo /etc/services.

O comando na linha 14: “ *x=`echo $b |sed "s/\//:/"|awk -F: '{print "iptables -A FORWARD -i br0 -p " $2 " --sport " $1 " -j ACCEPT" }`* ” , significa: echo (imprima) o campo $b |*sed "s/\//:/ ,* substituindo a \ por “ : ”. Na mesma linha use o comando |awk –F:, buscando o separador “ : ” e imprima $2 (porta) e $1 (número da porta).

O mesmo procedimento é feito para a variável y, que é para registro do tráfego de saída.

Na linha 22 “ *iptables -A FORWARD -j ACCEPT* “ , é usado o comando iptables genérico para portas desconhecidas, ou seja, que não fazem parte dos arquivos /etc/services e /etc/

Na última linha “ *rm -f /tmp/regras* “ é apagado o arquivo de controle *regras*.

4.2.6.2 Criação da tabela PROJETO

Para geração gráfica das oito aplicações foi criada a tabela “projeto”, cujos parâmetros do banco SQLITE foram definidos:

create table projeto (id INTEGER PRIMARY KEY AUTOINCREMENT ,proto char(15), dthr datetime, bytes char(15), pacotes char(15), sentido char(1));

create index idx_proto ON projeto(proto);
create index idx_dthr ON projeto(dthr);
create index idx_bytes ON projeto(bytes);
create index idx_pacotes ON projeto(pacotes);
create index idx_sentido ON projeto(sentido);

Que significa: cria a tabela projeto, utilizando a opção de alimentação de dados INTEGER, chave primária com auto incremento, onde foram criadas as colunas sucessivamente da primeira até a quinta: protocolo, dthr (datahora), bytes, pacotes, sentido.

Nas linhas seguintes estão definidas as opções de index (indexação) para cada tabela.

4.2.6.3 O script responsável pela coleta de contadores das oito aplicações – o rrdupdate.sh

Em resumo este script é script mais complexo do projeto, pois tem a função de buscar as oito aplicações, e atualizar o consumo de tráfego a partir dos contadores do iptables. Ele utiliza como base, o scripts iptables2.sh

```
#!/opt/bin/bash
local="192.168.2.0"
rrd=/opt/bin/rrdtool
db="/opt/projeto/projeto"
sql="/tmp/projeto.sql"
sqlite="/opt/bin/sqlite"
#14      644 ACCEPT     tcp  --  br0    any     anywhere             anywhere             tcp dpt:www
#12      552 ACCEPT     tcp  --  br0    any     anywhere             anywhere             tcp dpt:webcache
#2577  1108013 ACCEPT     0    --  br0    any     anywhere             anywhere             LAYER7 l7proto
finger

#filtro aplicacoes entrando...
echo filtro aplicacoes conhecidas
iptables -L FORWARD -v -x |grep "br0.*any.*anywhere.*spt" |sed "s/:/ /"  | grep -v "0 .*0.*ACCEPT" >
/tmp/iptables.i.tmp
echo filtro aplicacoes l7
iptables -L FORWARD -v -x |grep "br0.*any.*anywhere.*$local" | grep -v "0 .*0.*ACCEPT" >>
/tmp/iptables.i.tmp

#filtro aplicacoes saindo...
echo filtro aplicacoes conhecidas e aplicacoes l7
iptables -L FORWARD -v -x | grep "any.*br0" |sed "s/:/ /" |sort -n -r -k 2 | grep -v "0 .*0.*ACCEPT"  >
/tmp/iptables.o.tmp

#Limpa a tabela do Iptables
iptables -Z
for z in i o
do
basedir="/opt/share/www/top8"
dir="$basedir/$z"
echo Gerando e enviando os arquivos .rrd de entrada para posterior geracao grafica, a partir do diretorio:
$dir
echo ".mode list" > $sql
echo ".output stdout" >> $sql
cat /tmp/iptables.$z.tmp |while read a b c d e f g h i j k l
# 87    96826 ACCEPT     tcp  --  br0    any     anywhere             anywhere             tcp spt https
# 334    25551 ACCEPT     udp  --  br0    any     anywhere             anywhere             udp spt snmp
# 0       0 ACCEPT     0    --  any    br0     172.16.7.0/24        anywhere             LAYER7 l7proto teamspeak
# 0       0 ACCEPT     0    --  any    br0     172.16.7.0/24        anywhere             LAYER7 l7proto
teamfortress2
```

```
 do
 if [ ${#l} -eq 0 ];then
  l="outros"
 fi
 #Verifica se houve trafego no protocolo
 if [ $a -gt 0 ];then
 #verifica a posicao das colunas, e fornece os parametros no sql para posterior gravacao no
banco
  if [ $f = br0 ];then
      echo "insert into projeto (proto,tipo,dthr,bytes,pacotes,sentido) values
('$l','$j',strftime('%s','now'),$b,$a,'$z');" >> $sql
  else
      echo "insert into projeto (proto,tipo,dthr,bytes,pacotes,sentido) values
('$l','$j',strftime('%s','now'),$b,$a,'$z');" >> $sql
  fi
 fi
 done
echo ".quit" >> $sql
#GRAVA NO BANCO OS VALORES LIDOS
touch /tmp/db.locked
while [ -f /tmp/db.locked ]
 do
 $sqlite $db < $sql  && rm -f /tmp/db.locked
 sleep 1
 done

#Apaga resultado Anterior
rm -f /tmp/top8.$z.sql

#GERA O SQL INTERMEDIARIO COM OS TOTAIS PARCIAIS E LISTA DE PROTOCOLOS
touch /tmp/db.locked
while [ -f /tmp/db.locked ]
        do
 $sqlite $db < /opt/projeto/resultado.$z.sql | grep "^select.proto" > /tmp/top8.$z.sql && rm -f
/tmp/db.locked
 sleep 1
 done
$sqlite $db < /tmp/top8.$z.sql | sed "s/|/ /g" | while read l d c b
 do
 #    kb parcial       kb total
 # l     d        c        b
 #smb 137636812 141952 147676741
 #ftp 29625058 29354 29625058
 echo $l $d
 l=`echo $l | sed s/-/_/`
 if [ ! -f $dir/$l.rrd ];then
      echo -n `date` $l.rrd nao existe,criando...
      #rotina de criacao do rrd
      $rrd create $dir/$l.rrd --step 60 \
      DS:$l:COUNTER:120:U:U \
      RRA:AVERAGE:0.5:1:864  \
      RRA:AVERAGE:0.5:1:120
            echo ok
 fi
 #rotina de atualizar o rrd
        bit=8
 valor0=`expr $b \* $bit`
 echo -n `date` atualizando o rrd para $l ...
 $rrd update $dir/$l.rrd N:$valor0
 echo ok

 if [ -f $dir/$l.png ];then
  echo "Excluindo grafico anterior";
  rm -f $dir/$l.png
```

```
 fi

 done
done
```

4.2.6.4 O script responsável pela geração gráfica das oito aplicações - grafico.php

Este script possui funcionalidade para geração gráfica instantânea de aplicações trafegadas na bridge. Quando o administrador clicar no ícone “Visualizar Gráfico RRD”, será aberta uma janela trazendo informações gráficas de tráfego para a aplicação especifica que ele solicitou através desse script, em conjunto com o script rrdgraph.sh.

```
<html>
<head>
<meta http-equiv="refresh" content="100"; URL="grafico.php?proto=<?echo
$_GET['proto']."&sentido=".$_GET['sentido'];?>" >
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META HTTP-EQUIV="Expires" CONTENT="-1">
<title> Projeto Eng. Computacao </title>
</head>
<body>
<font size="2" face="Verdana">
<b>Analise de Trafego de Aplicacoes e Priorizacao de Pacotes em redes TCP/IP.</b>
</font>
<br>
<br>

<?php
include ("lib/lib.php");

$filename = "/opt/share/www/top8/$sentido/$proto.png";

if (! file_exists($filename))
        {
        $return=system ("/opt/projeto/rrdgraph.sh $proto $sentido");
        }
?>
<CENTER>
<img src="<?echo "/top8/$sentido/$proto.png";?>" alt="Coletando estatisticas, aguarde 1min..." />
</CENTER>

root@DD-WRT:/opt/share/www# cat /opt/projeto/rrdgraph.sh
#!/opt/bin/bash
prefix="/opt/share/www/top8/";
rrd="/opt/bin/rrdtool";
dir=$prefix$2;
l=`echo $1| sed s/-/_/g`;
#echo -n `date` atualizando o grafico para $l ...
$rrd graph $dir/$l.png -a PNG -t "Top 8 - Aplicacao ($l)" --vertical-label 'bits por segundo' \
        DEF:$l=$dir/$l.rrd:$l:AVERAGE \
        VDEF:"$l"max="$l",MAXIMUM \
        VDEF:"$l"avg="$l",AVERAGE \
        "CDEF:"$l"bits="$l",8,*" \
        --start end-20min \
        AREA:"$l"#FF0000:"$l" \
        COMMENT:"Maxima " \
        GPRINT:"$l"max:"%6.2lf %Sbps" \
        COMMENT:"Media " \
        GPRINT:"$l"avg:"%6.2lf %Sbps\l"
```

4.2.7 O script para controle de tráfego - qdisc.sh

Foi projeto foi simulado um enlace de dados de 1 Mbps. Este script define as filas (qdisc) e as classes (class) contendo 4 tipos:

- Máxima de 60% a 100% da velocidade da porta (1024 Kbps) com prioridade 0;
- Alta de 40% a 100% da velocidade da porta com prioridade 1;
- Média de 20% a 100% da velocidade da porta com prioridade 2;
- Baixa de 6% a 100% da velocidade da porta com prioridade 3.

Abaixo o script qdisc.sh:

```
#Disciplinas de Filas - QoS

#delete root qdisc (this will destroy all classes)
tc qdisc del dev vlan1 root
tc qdisc del dev vlan0 root
tc qdisc del dev imq0 root  #qdisc padroes do dd-wrt
tc qdisc del dev br0 root   #qdisc padroes do dd-wrt

#DEFININDO POLICY INBOUND

#attach root qdisc and create the 100Mbps root class
tc qdisc add dev vlan1 root handle 2: htb default 12
tc class add dev vlan1 parent 2: classid 2:10 htb rate 1024Kbit ceil 1024Kbit

#create the 384Kbps class for the whole bandwidth
#tc class add dev vlan1 parent 2:10 classid 2:20 htb rate 384Kbit

#maxima - 60% - 100% - pri 0 (pri2)
tc class add dev vlan1 parent 2:10 classid 2:100 htb rate 614Kbit ceil 1024kbit pri 0
tc qdisc add dev vlan1 parent 2:100 sfq quantum 1514b perturb 15
tc filter add dev vlan1 protocol ip parent 2:0 prio 1 handle 2 fw classid 2:100

#alta - 40% (409.6) - 100% - pri1 (pri3)
tc class add dev vlan1 parent 2:10 classid 2:200 htb rate 409Kbit ceil 409Kbit pri 1
tc qdisc add dev vlan1 parent 2:200 sfq quantum 1514b perturb 15
tc filter add dev vlan1 protocol ip parent 2:0 prio 1 handle 3 fw classid 2:200

#media - 20% (204.8) - 100% - pri2 (pri4)
tc class add dev vlan1 parent 2:10 classid 2:300 htb rate 204Kbit ceil 204Kbit pri 2
tc qdisc add dev vlan1 parent 2:300 sfq quantum 1514b perturb 15
tc filter add dev vlan1 protocol ip parent 2:0 prio 1 handle 4 fw classid 2:300

#baixa - 6%  (61.4) - 100% - pri3 (pri5)
tc class add dev vlan1 parent 2:10 classid 2:400 htb rate 61Kbit ceil 61Kbit pri 3
tc qdisc add dev vlan1 parent 2:400 sfq quantum 1514b perturb 15
tc filter add dev vlan1 protocol ip parent 2:0 prio 1 handle 5 fw classid 2:400

#DEFININDO POLICY OUTBOUND

#attach root qdisc and create the 100Mbps root class
tc qdisc add dev vlan0 root handle 2: htb default 12
tc class add dev vlan0 parent 2: classid 2:10 htb rate 1024Kbit ceil 1024Kbit

#tc qdisc add dev vlan0 root handle 2: htb
#tc class add dev vlan0 parent 2:0 classid 2:10 htb rate 100Mbit

#create the 384Kbps class for the whole bandwidth
#tc class add dev vlan0 parent 2:10 classid 2:20 htb rate 384Kbit

#maxima - 60% - 100% - pri 0 (pri2)
tc class add dev vlan0 parent 2:10 classid 2:100 htb rate 614Kbit ceil 1024Kbit pri 0
tc qdisc add dev vlan0 parent 2:100 sfq quantum 1514b perturb 15
tc filter add dev vlan0 protocol ip parent 2:0 prio 1 handle 2 fw classid 2:100
```

```
#alta - 40% (409.6) - 100% - pri1 (pri3)
tc class add dev vlan0 parent 2:10 classid 2:200 htb rate 409Kbit ceil 409Kbit pri 1
tc qdisc add dev vlan0 parent 2:200 sfq quantum 1514b perturb 15
tc filter add dev vlan0 protocol ip parent 2:0 prio 1 handle 3 fw classid 2:200

#media - 20% (204.8) - 100% - pri2 (pri4)
tc class add dev vlan0 parent 2:10 classid 2:300 htb rate 204Kbit ceil 204Kbit pri 2
tc qdisc add dev vlan0 parent 2:300 sfq quantum 1514b perturb 15
tc filter add dev vlan0 protocol ip parent 2:0 prio 1 handle 4 fw classid 2:300

#baixa - 6%  (61.4) - 100% - pri3 (pri5)
tc class add dev vlan0 parent 2:10 classid 2:400 htb rate 61Kbit ceil 61Kbit pri 3
tc qdisc add dev vlan0 parent 2:400 sfq quantum 1514b perturb 15
tc filter add dev vlan0 protocol ip parent 2:0 prio 1 handle 5 fw classid 2:400
```

4.2.7.1 O script engine_qos.php para aplicação de QoS das aplicações.

```
<?
include("lib/lib.php");

//$db->debug=true ;
#var_dump($_POST);
################################################################################
#  Update
if(isset($modo)&&$modo=="update")
{
# Tratamento do Variaveis
if ($prioridade > 0 and $proto != "" )
 {
 #$prioridade=$_POST["prioridade"];
 #$proto=$_POST["proto"];
 #$tipo=$_POST["tipo"];
 $db->Execute("update qos set prioridade=".$prioridade." where proto='".$proto."' and tipo ='".$tipo."'");
 #rotina para excluir as regra do mangle
 $saida=system ("/usr/sbin/iptables -F -t mangle",$retval);
 #rotina para incluir a nova regra do mangle
 $rs =$db->Execute("select proto,tipo,prioridade from qos where prioridade > 1");
 $TotalRegistros= $rs->RecordCount();
 while (!$rs->EOF) {
  if ( $rs->fields['tipo'] == "LAYER7")
   {
   $iptables = "/usr/sbin/iptables -t mangle -A FORWARD -i vlan1 -m layer7 --l7proto  ". $rs-
>fields['proto']." -j MARK --set-mark  ".$rs->fields['prioridade']."";
   }
  else
   {
   $iptables = "/usr/sbin/iptables -t mangle -A FORWARD -i vlan1 -p  ". $rs->fields['tipo'] ." --sport  ". $rs-
>fields['proto'] ." -j MARK --set-mark  ". $rs->fields['prioridade']."";
   }
  #print $iptables ."<br>";
  system ($iptables);
              $rs->MoveNext();
              }
 }
}
cabecalho();
?>
<br>
<a><font size=2>
Regras aplicadas com sucesso
</font></a>
<script>window.opener.reload();</script>
```

CAPÍTULO 5 – APLICAÇÃO DA SOLUÇÃO COM RESULTADOS

Este capítulo inicia com descrição sucinta do local onde será realizada a aplicação total da solução, informando o cenário e as condições. Inclui a descrição de como é feita a demonstração a banca examinadora evidenciando as funcionalidades da ferramenta proposta. Termina com uma pesquisa analítica explicando a razão de se escolher oito aplicações para demonstração gráfica.

5.1 Apresentação do ambiente de simulação e implementação do modelo proposto

Na apresentação do modelo proposto, serão disponibilizados os seguintes equipamentos:

- 1 (um) Roteador Cisco 2611Xm, composto por duas portas Fast Ethernet;
- 2 (dois) endpoints Sony Ipela (equipamentos - Video conferência);
- 1 (uma) bridge (projeto) D-link, modelo DIR-320;
- 2 (dois) switchs de rede;
- 2 (dois) notebooks;
- Cabos de rede e de energia necessários;

A figura 08 ilustra a topologia empregada no ambiente de simulação do projeto:

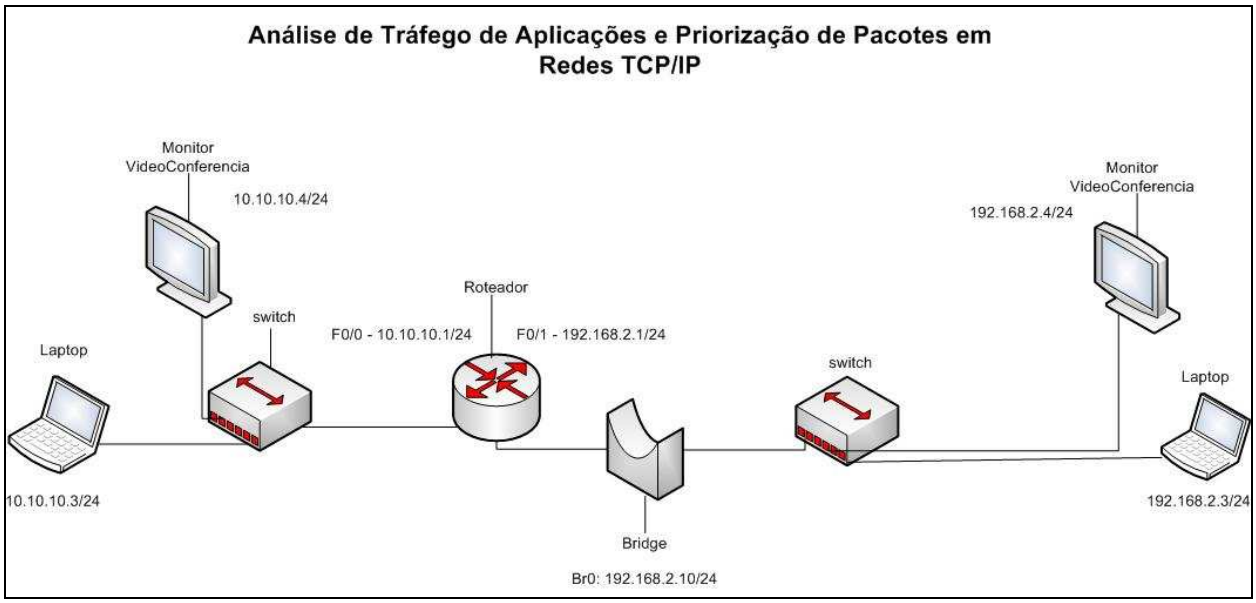

Fig. 08 - Topologia empregada no ambiente de simulação do projeto

Na apresentação da implementação da solução, todos os equipamentos serão montados em uma sala a ser disponibilizada pelo UniCEUB, possuindo infra-estrutura preparada para comportar a energização dos equipamentos operando em voltagem de 110V AC. Todos os cabos de rede necessários a interligação dos equipamentos, serão disponibilizados e providos por este aluno.

A demonstração do projeto consiste em simular um ambiente em produção com umas das aplicações mais utilizadas em uma rede coorporativa, dentre elas estão: um servidor de arquivos (samba), um servidor web (http), e uma conferência por vídeo (h323).

Um dos notebooks localizado em uma das subredes iniciará o processo de download de um arquivo via http através de outro notebook (simulando um servidor web - *webserver*) localizado na outra subrede.

Será simulado um enlace de dados wan, sendo um circuito dedicado com velocidade de 1024 Kbps.

Ao iniciar o download do arquivo, a banca examinadora poderá acompanhar via interface gráfica, o consumo de banda (throughput) da aplicação, sendo que, no mesmo instante, será disparada uma seção de videoconferência entre dois monitores, cada um localizado em uma subrede. A visualização de consumo de banda desta aplicação também poderá ser checada.

Nesta etapa, todo o tráfego da rede que passa pela bridge não será aplicado qualidade de serviço, simulando um tráfego desordenado. Nesta situação é perceptível a degradação da qualidade da transmissão da videoconferência, com características de congelamentos instantâneos, travamentos da voz, e delay, pois ocorre no mesmo instante, conforme informado, transferência de arquivos através de servidor Web.

Para obter um padrão de perfeição na transmissão da videoconferência, será aplicado qualidade de serviço (QoS) no tráfego, implementado através da bridge, aplicado pelo administrador via interface web.

Na configuração do QoS serão priorizadas as aplicações rtp (real time protocol), e H.323, que são protocolos utilizados nesse tipo de transmissão. No mesmo instante será aplicada baixa prioridade para a aplicação http.

A mudança da qualidade da transmissão da videoconferência estará perceptível dentro de alguns segundos, devido ao benefício da implementação da qualidade de serviço no tráfego.

5.2 Descrição da Aplicação da Solução

Analisando de forma geral o desenvolvimento deste projeto ao longo do semestre, em sua etapa inicial, foi dado início a pesquisa de como gerar os gráficos. Foi descoberta uma ferramenta ideal para atingir esse objetivo, chamada de *RRDTOOL*, criada em 1999, por Tobias Oitiker.

A partir desse momento, foram estudadas estratégias para coleta, e atualização dos dados a serem carregados pela ferramenta para o propósito final, que seria a geração dos gráficos. Os dados a serem coletados vieram do registro dos pacotes ip (modelo TCP/IP), onde no instante que um pacote ip passava (FORWARD) pela bridge era contabilizado *(match) no IPTABLES* através de regras definidas baseadas na porta de conexão (camada de transporte). Para registro de pacotes que utilizavam portas dinâmicas, exemplo: FTP, RTP, E-DONKEY, houve a necessidade de identificação destes pacotes. Para isso foi utilizado a pacote *L7-filter*, compilado ao *IPTABLES*. Sua implementação foi bastante satisfatória, sendo um dos diferenciais do projeto.

No decorrer do desenvolvimento do projeto, foi analisada uma questão: por que invés de instalar o projeto no notebook que possui uma configuração de hardware peculiar, não tornar a instalação de todo o projeto unificada para rodar num determinado equipamento que possa ter a mesma configuração de hardware? Por que não proporcionar uma maneira unificada para reparar a configuração, bastando apenas substituir a configuração por outro de mesma marca e modelo?

Em contato com profissionais de TI, foi descoberto que a comunidade Linux havia desenvolvido um firmware baseado no kernel 2.4 para rodar no equipamento roteador wireless D-Link DIR-320 [www.dd-wrt.com]. Tendo essa informação, a motivação foi maior, pois atenderia a idéia de unificar o projeto em um equipamento que tivesse o mesmo hardware, com vantagem no suporte ao carregamento de configuração (upload) em outro equipamento caso o primeiro estivesse defeituoso, além de montagem do sistema de arquivos (ext 2) a partir do pendrive, incluindo fácil instalação física da bridge, onde as portas podem ser visualmente identificadas (LAN e WAN) caso um terceiro seja instruído a instalar o equipamento em uma rede filial geograficamente distribuída, além do backup da configuração do equipamento, e por último a vantagem de possuir interfaces físicas de rede de backup. Dessa forma, o projeto foi transferido do modelo inicial - notebook, para o roteador D-link DIR-320.

Partindo para a busca das 8 aplicações mais trafegadas da rede, cabe ressaltar que foi uma das etapas mais difíceis do projeto, foi desenvolvido um script que iniciava a busca através de vetores, comparando os maiores valores de bytes provenientes para cada regra do *IPTABLES*. Nessa etapa foi encontrada mais uma dificuldade: na geração do gráficos; A dificuldade estava no momento da visualização do gráficos gerados a cada 20 min., como na seguinte situação: A transferência de um arquivo utilizando o protocolo FTP foi realizada às 7h20 de um dia qualquer sendo registrada no iptables. O script que faz a busca das oito aplicações mais trafegadas é executado. Logo, os gráficos das oito aplicações começam a ser construídos, incluindo a primeira aplicação mais trafegada, o FTP. No decorrer de 20 min, o FTP aparece como primeiro. Após 31 min às 7h51 mesmo depois da transferência ter acabado às 7h40, o FTP continua sendo mostrado como a aplicação mais trafegada, e no gráfico não há informação de tráfego em Kbps para a aplicação FTP. O problema estava aí, ou seja, no script o vetor de comparação sempre trazia o protocolo FTP como sendo a maior aplicação trafegada, só que mesmo após terem se passados 20 min, continuava aparecendo como a maior, só que não havia mais nenhum tráfego para essa aplicação.

Essa falha foi corrigida com a utilização do banco SQLITE. O script foi refeito, para que a busca das oito aplicações se tornasse dinâmica com a ajuda fundamental do banco de dados. Sua função foi armazenar os dados apenas dos 20 minutos corridos. Após esse período, era novamente carregado em sincronia à medida que gráfico estava sendo construído. Ou seja, sua tabela era renovada a cada 20 minutos.

O gráfico 01 ilustra o Throughput do Link durante 20 min e os gráficos 02 a 09 ilustram a velocidade em Kbps das oito aplicações mais trafegadas em uma rede.

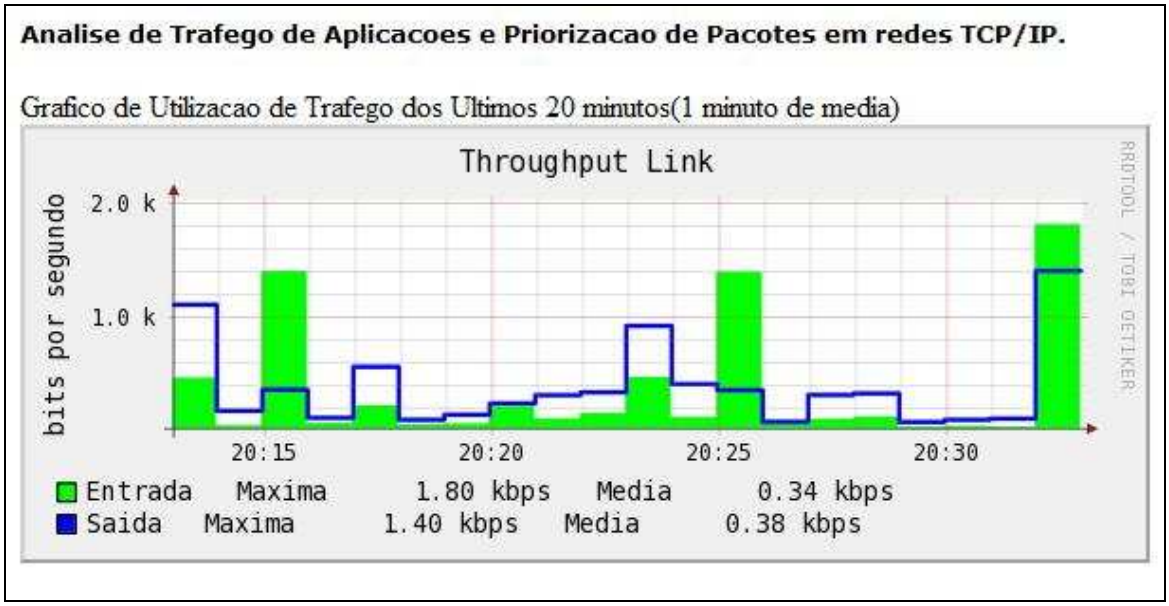

Gráfico 01 - Throughput do Link durante 20 min

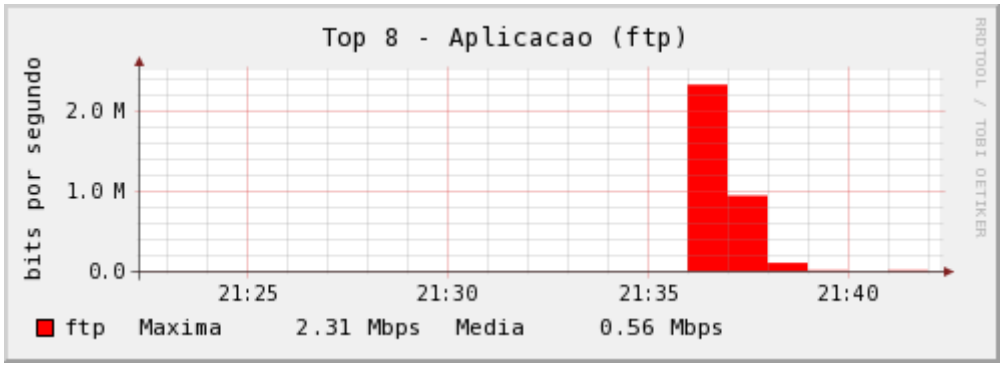

Gráfico 02 - Utilização do protocolo FTP

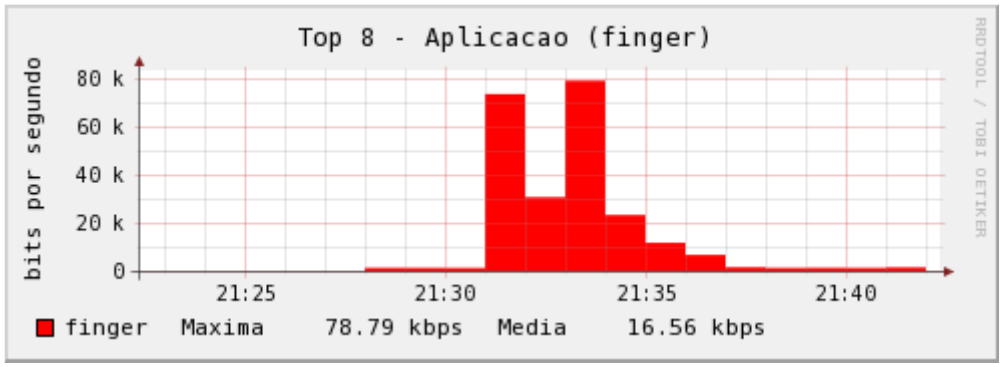

Gráfico 03 - Utilização do protocolo finger

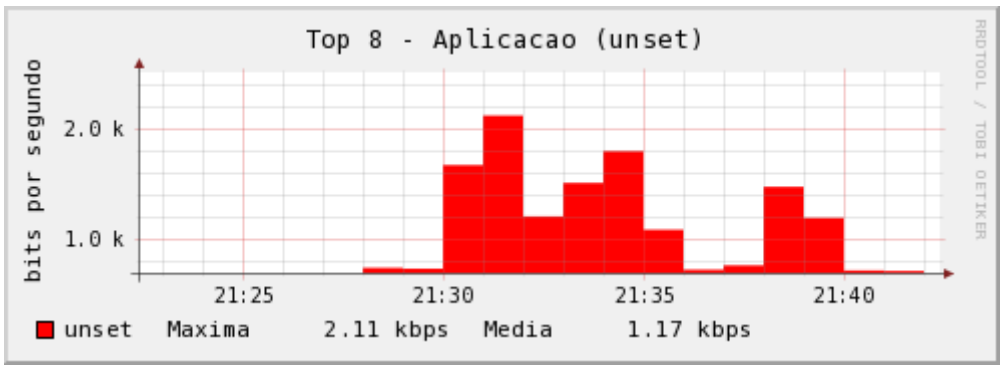

Gráfico 04 - Utilização do protocolo unset

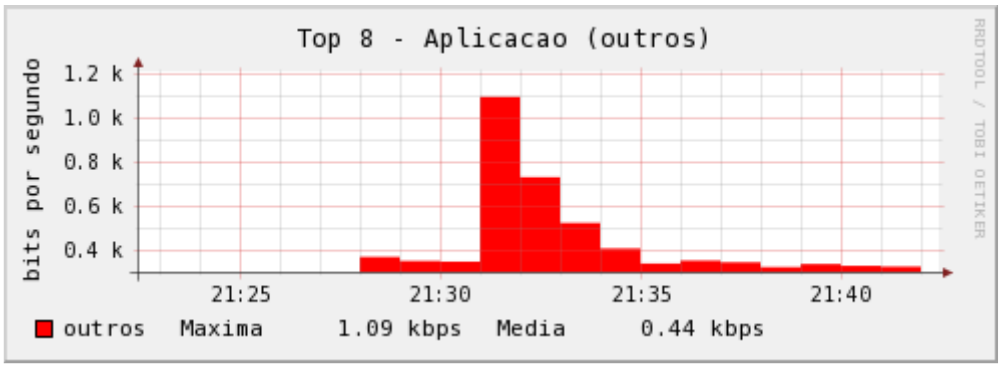

Gráfico 05 – Utilização do protocolo (outros)

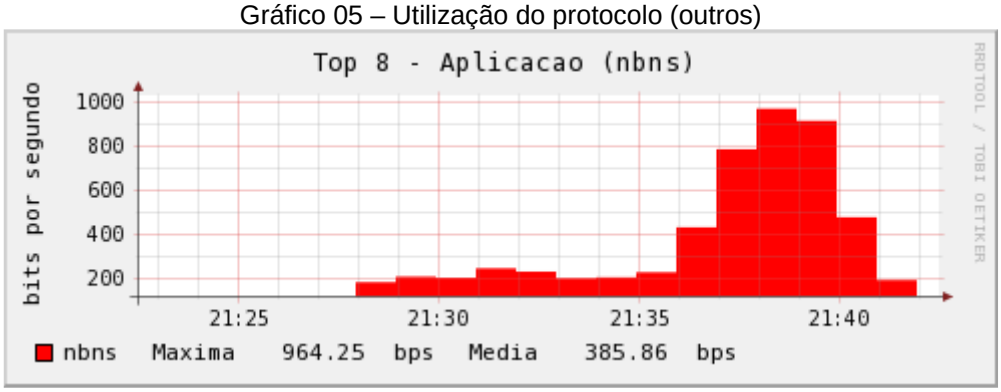

Gráfico 06 - Utilização do protocolo nbns

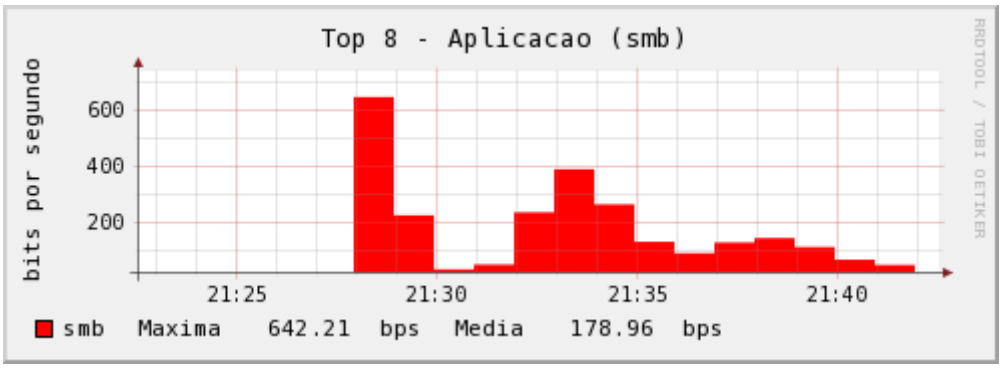

Gráfico 07 - Utilização do protocolo smb

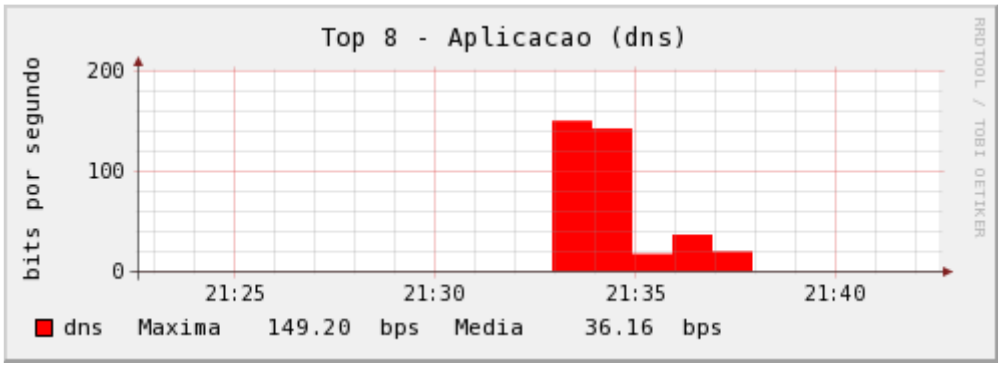

Gráfico 08 - Utilização do protocolo dns

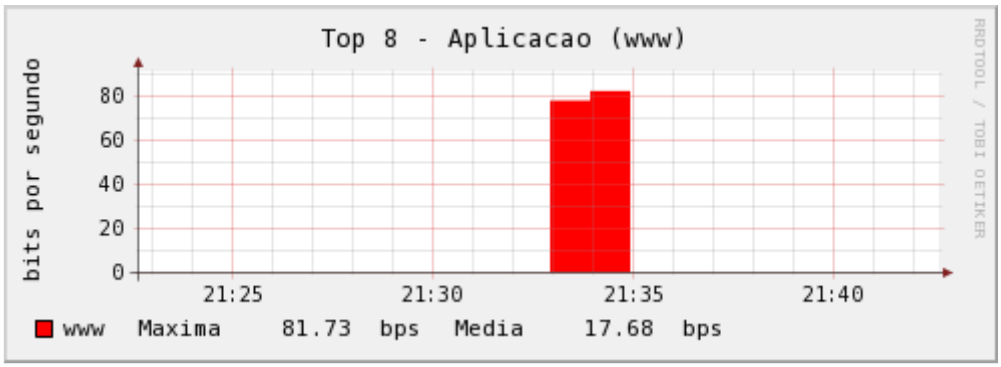

Gráfico 09 - Utilização do protocolo www

Na implementação da qualidade de serviço (qos), faz-se a observação que se deve aplicá-lo tanto na interface de saída (interface wan), quanto na interface de entrada (interface lan), pois o resultado não foi satisfatório aplicando somente a priorização do tráfego em uma das interfaces. A configuração de qualidade de serviço deve ser feita de forma simétrica, ou seja, as regras aplicadas são feitas de forma idêntica tanto na interface de entrada, quanto na interface de saída, na bridge.

A figura 09 ilustra a interface web para aplicação de qualidade de serviço na bridge.

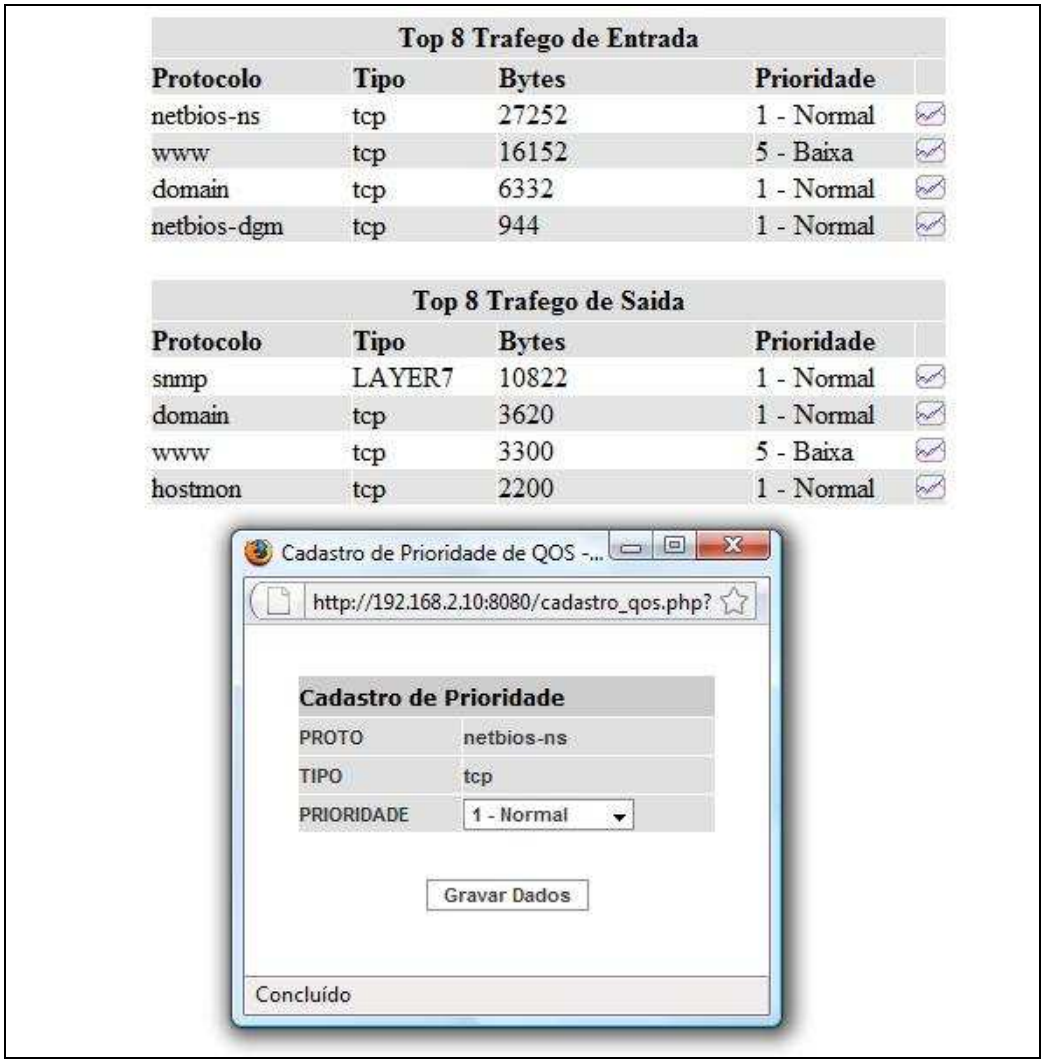

Fig. 09 – Interface web para aplicação de qos

5.3 Avaliação global do momento de solução proposto

Analisando o escopo inicial deste projeto, onde o equipamento proposto na sua implementação seria um notebook, houve uma melhora significativa no projeto, substituindo-o por um roteador, que possui a vantagem de ter memória flash, menos suscetível a erros, em comparação ao disco magnético. Outras vantagens foram:

- fácil instalação tanto física quanto lógica;
- suporte a backup e upload de configuração;
- armazenamento em sistema de arquivo ext2 a partir de pendrive utilizando memória flash;
- interfaces físicas de rede de backup.

A desvantagem na utilização deste hardware foi apenas uma: o limite de processamento e memória. Possuindo um processador MIPS com 240 MHz, e apenas 32 MB de RAM, foram encontradas dificuldades na implementação do projeto, sendo encaradas através de uma percepção meticulosa na análise do problema e consequentemente na busca da melhor solução. Dessa forma, foram adicionados controles para execução dos processos de forma ordenada a fim de, não sobrecarregar o hardware, tão limitado.

O registro de informações de pacotes utilizando a ferramenta iptables, teve interação direta com o banco de dados, ferramenta que teve papel fundamental para aliviar o processamento da bridge.

Nota-se que durante os testes do fluxo de pacotes, gerando os gráficos, implementando qos, ambos de forma simultânea, a bridge não conseguiu processar suas funcionalidades com tráfego acima de 2Mbps. Dessa forma, não é indicado a utilização deste equipamento para enlaces de dados acima de 2Mbps.

Em um contexto geral, os resultados foram satisfatórios respeitando de certa forma a limitação do equipamento, pois implementar todas as funcionalidades descritas neste projeto num equipamento com pouco recurso de hardware, torna este projeto um grande desafio pessoal e profissional.

5.4 Análise de tráfego de aplicações em Redes Corporativas

No início do desenvolvimento deste projeto, a fim de definir diretrizes para construção dos gráficos, foi feita pesquisa dos tipos de aplicações que mais consomem a banda num Link wan.

A empresa pesquisada utiliza tecnologia MPLS com velocidade na porta do roteador de 1024 Kbps simétrico (taxas de download e upload iguais).

No estudo, consta relatório das 8 (oito) aplicações mais trafegadas no mês de fevereiro, cujas taxas de entrada (Inbound) e de saída (Outbound) estão discriminadas, conforme tabelas 1 e 2:

Top 20 Inbound Applications					
	Application	Packets	Data (MB)	Throughput Avg (kbps)	Throughput Max (kbps)
1	CIFS	57196064	9983.294	1.31	890.46
2	HTTP	8575413	5974.523	6.78	1748.57
3	RDP	51245073	4067.311	3.36	366.16
4	Windows Updates	1160581	1385.404	41.03	793.22
5	Sites Gov	1114818	1078.279	15.39	1640.79
6	MAPI	1080723	923.516	12.20	303.69
7	ExindaWM	6232197	512.159	0.23	14.70
8	FTP	339372	438.715	213.67	2840.36
9	HTTPS	785777	426.205	4.54	695.99
10	SAV	431227	424.030	34.88	188.68
11	Quicktime	216960	306.299	164.11	698.94
12	SIP	1293632	276.168	7.65	80.78
13	Sites Jus	197639	200.741	14.99	1000.65
14	ExindaCom	2305613	183.486	0.83	17.64

Tabela 1 - Análise de tráfego de aplicações para entrada de um Link corporativo de 1024 Kbps (MPLS).
Fonte: TRF da 1ª. Região

Top 20 Outbound Applications					
	Application	Packets	Data (MB)	Throughput Avg (kbps)	Throughput Max (kbps)
1	CIFS	40423251	6594.676	0.87	1877.08
2	RDP	44165301	2801.264	2.31	190.52
3	HTTP	7291132	1447.106	1.63	249.54
4	SIP	1573932	422.643	5.71	78.33
5	HTTPS	769575	275.317	2.94	354.93
6	MAPI	961894	196.629	2.60	281.36
7	AD Replication	358604	158.783	4.96	611.63
8	RTP-Sony	799157	152.341	64.50	78.37
9	ssdp	406881	134.114	1.96	2.74
10	Sites Gov	850230	108.119	1.54	151.97
11	squidproxy	199045	67.797	1.69	90.99
12	Windows Updates	643053	43.297	1.28	12.88
13	DHCP	54932	24.822	2.66	26.29
14	SAV	276556	23.713	1.94	8.75

Tabela 2 - Análise de tráfego de aplicações para saída de um Link corporativo de 1024 Kbps (MPLS).
Fonte: TRF da 1ª. Região

Segundo informações da tabela 1 as oito aplicações mais trafegadas na entrada são: CIFS, HTTP, RDP, Windows updates, Sites Gov, MAPI, Exinda WM, FTP. Para saída, segundo tabela 2, as oito aplicações mais trafegadas são: CIFS, RDP, HTTP, SIP, HTTPS, MAPI, AD Replication, RTP-Sony.

Para o projeto a escolha da visualização gráfica de 8 (oito) aplicações foi definida de forma estratégica (com base analítica), pois é essa a quantidade que demanda mais tráfego. Dessa forma, o administrador pode aplicar o QoS baseando-se na análise gráfica das oito aplicações que mais consomem banda, priorizando ou minimizando cada uma.

CAPÍTULO 6 – CONCLUSÃO

A proposta de implementação de uma ferramenta para análise de tráfego, assim como a de proporcionar a aplicação de qualidade de serviço a partir de interface web, em um equipamento limitado tanto em processamento, quanto em memória tornou o desenvolvimento deste projeto um desafio árduo. Dificuldades foram enfrentadas, novos conceitos foram adquiridos, como o uso do banco de dados e da interface web.

A flexibilidade no uso das ferramentas que compõem este projeto, o tornam potencialmente atrativo possuindo recursos como: construção gráfica de aplicações identificadas em nível de aplicação em camada 7, aplicação de qos em aplicações da camada 7, visualização da velocidade real da enlace de dados, fácil instalação e manutenção, pois é um hardware específico para um mesmo modelo de equipamento.

A ferramenta desenvolvida neste projeto mostrou-se um facilitador no gerenciamento da rede , já que possibilita ao administrador um melhor conhecimento do tráfego de entrada e saída , portanto, é um recurso a mais para identificar possíveis falhas e a consequente solução das mesmas, através da aplicação de QoS.

6.1 Sugestão para trabalhos futuros

A sugestão de complemento deste trabalho é a expansão da visualização das aplicações para um período maior de uma semana, ou um mês, habilitando histórico de tráfego neste período.

Outro complemento seria a geração gráfica para visualização de todas as aplicações que são priorizadas pelo QoS. Dessa forma, será possível ver os níveis percentuais de utilização da banda através das filas aplicadas e os respectivos consumos das mesmas.

REFERÊNCIAS BIBLIOGRÁFICAS

BAETA, Márcio Pachioni. *Curso de Introdução às Redes de Computadores*. Brasília, 2001.

FERREIRA, Rubem E. *Linux: Guia do Administrador do Sistema.* Novatec, 2003.

GHEORGHE, Lucian. *Designing and Implementing Linux Firewall and QoS using netfilter, iproute2, NAT, and L7-filter*. Birmingham: Packt Publishing Ltd., 2006

GIL, Antônio Carlos. *Como elaborar projetos de Pesquisa*. 4. ed. São Paulo: Atlas, 2002.

GOTTSCHALL, Sebastian. *SOBRE O DD-WRT*. Alemanha, 2009. Disponível em: <http://www.dd-wrt.com/dd-wrtv3/dd-wrt/about.html>. Acesso em: 21 mar. 2009.

EMBRATEL, Empresa Brasileira de Telecomunicações. Site Oficial, 2003 - 2004. *Garantia de Desemplenho*. Disponível em: <https://webebt04.embratel.com.br/sgsla/cgi-bin/Natl_report_ultima_hora.pl> Acesso em: 10 abril 2009

HWACI. *Direitos Autorais, SQLITE.* Disponível em: <http://www.sqlite.org/copyright.html>. Acesso em: 5 maio. 2009.

LEVANDOSKI, Justin 2009. *Application Layer Packet Classifier for Linux*. Disponível em: <http://l7-filter.sourceforge.net>. Acesso em: 2 maio.2009

LOZANO, Fernando. *Licenças de Software Livre.* Disponível em: <http://www.gnu.org/licenses/licenses.pt-br.html>. Acesso em: 6 maio. 2009.

MARTINS, Fábio. *Implementação de um gateway de borda com controle de banda utilizando software livre.* Trabalho apresentado ao Centro Universitário de Brasília (UNICEUB) como pré-requisito para a obtenção de certificado de conclusão do curso de Engenharia da Computação. Brasília, 2005.

TANENBAUM, Andrew S. Redes *de Computadores*. 3 ed. Campus, 1997.

OETIKER, Tobias. *TUTORIAL DO RRDTOOL*. Suíça, 2009. Disponível em: <http://oss.oetiker.ch/rrdtool/tut/rrdtutorial.en.html>. Acesso em: 16 fev. 2009.

OLEO. O GERENCIADOR DE PACOTES OPTWARE, 2009. Disponível em: <http://www.dd-wrt.com/wiki/index.php/Optware>. Acesso em: 25 mar. 2009.

PURDY, Gregor N. *Linux Iptables: Guia de bolso.* Alta Books, 2005.

ANEXO 1

O script REGRAS

```
/bin/echo -n "INICIO " > /tmp/regras.log
/bin/date >> /tmp/regras.log

/bin/echo CONFIGURANDO A BRIDGE...
#adicionando a interface vlan1, vlan0 -> br0
/usr/sbin/brctl addif br0 vlan1
/usr/sbin/brctl addif br0 vlan0

#definindo ip na br0
#ifconfig br0 192.168.1.1 netmask 255.255.255.0 - ipdefault do DD-WRT
/sbin/ifconfig br0:2 172.16.7.253 netmask 255.255.252.0
/sbin/ifconfig br0:3 192.168.2.10 netmask 255.255.252.0
/sbin/ifconfig br0:4 172.18.253.140 netmask 255.255.0.0

#definindo rota
route add -net 0/0 gw 192.168.2.1 dev br0
#########################################################################

/bin/sleep 15
/bin/echo apagando os rrds para evitar decremento do trafego em comparação da hora atual...
rm /opt/share/www/top8/*.rrd
rm /opt/share/www/top8/*.png
rm /opt/share/www/top8/i/*.rrd
rm /opt/share/www/top8/i/*.png
rm /opt/share/www/top8/o/*.rrd
rm /opt/share/www/top8/o/*.png
rm /opt/share/www/rtime/*.rrd
rm /opt/share/www/rtime/*.png
/opt/projeto/rtime-20min.sh

#Limpa os dados de trafego do banco...
$sqlite $db < /opt/projeto/limpa_banco.sql

/bin/echo chamando iptables.sh...
/bin/sh /opt/projeto/iptables2.sh

echo Inicializando webserver...
/bin/sh /opt/etc/init.d/S80thttpd start

echo -n "FIM " >> /tmp/regras.log
/bin/date >> /tmp/regras.log
```

O Comando RRDTOOL

syntax disponível para o RRDTOOL:

```
RRDtool 1.2.15  Copyright 1997-2006 by Tobias Oetiker <tobi@oetiker.ch>
                Compiled Dec 14 2006 22:05:35

Usage: rrdtool [options] command command_options

Valid commands: create, update, updatev, graph, dump, restore,
                last, first, info, fetch, tune, resize, xport

RRDtool is distributed under the Terms of the GNU General
Public License Version 2. (www.gnu.org/copyleft/gpl.html)

For more information read the RRD manpages
```

O comando IPTABLES

Syntax disponível para o iptables

```
iptables v1.3.6

Usage: iptables -[AD] chain rule-specification [options]
       iptables -[RI] chain rulenum rule-specification [options]
       iptables -D chain rulenum [options]
       iptables -[LFZ] [chain] [options]
       iptables -[NX] chain
       iptables -E old-chain-name new-chain-name
       iptables -P chain target [options]
       iptables -h (print this help information)

Commands:
Either long or short options are allowed.
  --append  -A chain            Append to chain
  --delete  -D chain            Delete matching rule from chain
  --delete  -D chain rulenum
                                Delete rule rulenum (1 = first) from chain
  --insert  -I chain [rulenum]
                                Insert in chain as rulenum (default 1=first)
  --replace -R chain rulenum
                                Replace rule rulenum (1 = first) in chain
  --list    -L [chain]          List the rules in a chain or all chains
  --flush   -F [chain]          Delete all rules in  chain or all chains
  --zero    -Z [chain]          Zero counters in chain or all chains
  --new     -N chain            Create a new user-defined chain
  --delete-chain
           -X [chain]           Delete a user-defined chain
  --policy  -P chain target
                                Change policy on chain to target
  --rename-chain
           -E old-chain new-chain
                                Change chain name, (moving any references)
Options:
  --proto       -p [!] proto    protocol: by number or name, eg. `tcp'
  --source      -s [!] address[/mask]
                                source specification
  --destination -d [!] address[/mask]
                                destination specification
  --in-interface -i [!] input name[+]
                                network interface name ([+] for wildcard)
  --jump        -j target
                                target for rule (may load target extension)
  --goto      -g chain
                               jump to chain with no return
  --match       -m match
                                extended match (may load extension)
  --numeric     -n              numeric output of addresses and ports
  --out-interface -o [!] output name[+]
                                network interface name ([+] for wildcard)
  --table       -t table        table to manipulate (default: `filter')
  --verbose     -v              verbose mode
  --line-numbers                print line numbers when listing
  --exact       -x              expand numbers (display exact values)
[!] --fragment  -f              match second or further fragments only
  --modprobe=<command>          try to insert modules using this command
  --set-counters PKTS BYTES     set the counter during insert/append
[!] --version   -V              print package version.
```

O Comando TC

Syntax disponível para o comando tc

```
Usage: tc [ OPTIONS ] OBJECT { COMMAND | help }
       tc [-force] -batch file
where  OBJECT := { qdisc | class | filter | action | monitor }
       OPTIONS := { -s[tatistics] | -d[etails] | -r[aw] | -b[atch] [file] }
```

Script rtime-20min.sh

```
rtime=/opt/share/www/rtime
patch=/opt/bin/rrdtool

if ! [ -d $rtime ]
then
   mkdir -p /opt/share/www/rtime
fi

$patch create /opt/share/www/rtime/rtime-20min.rrd --step 60 \
DS:trafego_in_20min:COUNTER:120:U:U RRA:AVERAGE:0.5:1:864 \
DS:trafego_out_20min:COUNTER:120:U:U RRA:AVERAGE:0.5:1:864 \
```

Script rtime-at-20min.sh

```
#!/bin/sh
#sleep 15
echo /bin/date `/bin/date +%m%d%H%M%Y` > /jffs/opt/projeto/last_date.sh
ifconfig=/sbin/ifconfig
rrd=/opt/bin/rrdtool
path=/opt/share/www/rtime/rtime-20min.rrd
bit=8

valor_in=`$ifconfig vlan1 |awk -F: '/RX.bytes/{print $2--}'`
trafego_in=`expr $valor_in \* $bit`

valor_out=`$ifconfig vlan1 |awk -F: '/TX.bytes/{print $3--}'`
trafego_out=`expr $valor_out \* $bit`

$rrd update $path N:$trafego_in:$trafego_out

#--------------gerando o RRDGRAPH
rrd=/opt/bin/rrdtool
path=/opt/share/www/rtime

$rrd graph $path/rtime-20min.png -a PNG -t "Throughput Link" --vertical-label 'bits por segundo' \
DEF:trafego_in_20min=$path/rtime-20min.rrd:trafego_in_20min:AVERAGE \
DEF:trafego_out_20min=$path/rtime-20min.rrd:trafego_out_20min:AVERAGE \
VDEF:"trafego_in_20min"max=trafego_in_20min,MAXIMUM \
VDEF:"trafego_in_20min"avg=trafego_in_20min,AVERAGE \
VDEF:"trafego_out_20min"max=trafego_out_20min,MAXIMUM \
VDEF:"trafego_out_20min"avg=trafego_out_20min,AVERAGE \
--start end-20min \
--zoom 1.2 \
"CDEF:trafego_in_20min_bits=trafego_in_20min,8,*"  \
"CDEF:trafego_out_20min_bits=trafego_out_20min,8,*"  \
AREA:trafego_in_20min#00FF00:Entrada \
COMMENT:"Maxima " \
GPRINT:"trafego_in_20min"max:"%6.2lf %Sbps" \
COMMENT:"Media " \
GPRINT:"trafego_in_20min"avg:"%6.2lf %Sbps\l" \
LINE2:trafego_out_20min#0000FF:Saida \
COMMENT:"Maxima " \
GPRINT:"trafego_out_20min"max:"%6.2lf %Sbps" \
COMMENT:"Media " \
GPRINT:"trafego_out_20min"avg:"%6.2lf %Sbps\l"
```

Script top8.php

```
<html>
<head>
<meta http-equiv="refresh" content="3000"; URL="top8.php" >
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META HTTP-EQUIV="Expires" CONTENT="-1">
<title> Projeto Eng. Computacao </title>
</head>
<body>

<?
include("lib/lib.php");
#$db->debug=true;
##############################################################################
Cabecalho();
##############################################################################
AbreTabela();
##############################################################################
$sentidos=array('i','o');
$sentidos_desc=array('i'=>'Entrada', 'o'=>'Saida');
foreach ( $sentidos as $sentido)
 {
 #verifica os maiores protocolos dos ultimos 20 minutos
 $sql="select pj.proto as proto,sum(bytes) b,pd.desc as prioridade,qos.tipo as tipo from projeto pj
,qos,prioridade pd where pj.proto=qos.proto and qos.prioridade=pd.id and dthr > (strftime('%s','now') -
1200) and sentido='$sentido' group by pj.proto order by b desc limit 8 ";
 $Resultado=$db->execute($sql);
 $TotalRegistros= $Resultado->RecordCount();
 if($TotalRegistros==0)
        {
        echo "<center >Nao foi encontrado nenhum registro com os parametros
passados</center><br><br>";
        echo "<center ><a class=\"Link\" href=\"relatorio_vendasg.php\">NOVA CONSULTA</a></center>";
        }
 else
     {
     $_SESSION['rs_graph']=array();
     $i=0;
     ?>
     <table width="60%" border="0" cellpadding="0" cellspacing="1" align=center>
     <tr>
     <td height="23" bgcolor="#E0E0E0" class="Titulo1" colspan="5" <div align="center"><b>Top 8
Trafego de <?echo $sentidos_desc[$sentido];?></b></td>
     </tr>
     <tr>
     <td width="25%" bgcolor="#E0E0E0" class="TituloTd"><b>Protocolo</b></td>
     <td width="18%" bgcolor="#E0E0E0" class="TituloTd"><b>Tipo</b></td>
     <td width="32%" bgcolor="#E0E0E0" class="TituloTd"><b>Bytes</b></td>
     <td width="20%" bgcolor="#E0E0E0" class="TituloTd"><b>Prioridade</b></td>
     <td width="3%" bgcolor="#E0E0E0" class="TituloTd"></td>
     </tr>
     <?
     $corlinha="";
     while (!$Resultado->EOF)
  {
            $linha=$Resultado->GetRowAssoc();
            $linhap=preg_replace('/-/','_',$linha['PROTO']);
            $corlinha = ($corlinha == "#FFFFFF")?"#E4E4E4":"#FFFFFF";
  ?>
  <tr>
  <td width="25%" bgcolor="<?echo $corlinha?>" class="TituloTd"><a href="#"
onclick="JanelaCentralizada('cadastro_qos.php?p=<?echo $linha['PROTO'];?>',300,200);"
STYLE="color:#000000;text-decoration: none;"><font>
```

```
  <? echo  $linha['PROTO'] ?></a></font></td>
  <td width="18%" bgcolor="<?echo $corlinha?>" class="TituloTd"><a href="#"
onclick="JanelaCentralizada('cadastro_qos.php?p=<?echo $linha['PROTO'];?>',300,200);"
STYLE="color:#000000;text-decoration: none;"><font>
  <? echo  $linha['TIPO'] ?></a></font></td>
  <td width="18%" bgcolor="<?echo $corlinha?>" class="TituloTd"><a href="#"
onclick="JanelaCentralizada('cadastro_qos.php?p=<?echo $linha['PROTO'];?>',300,200);"
STYLE="color:#000000;text-decoration: none;"><font>
  <? echo  $linha['B'] ?></a></font></td>
  <td width="18%" bgcolor="<?echo $corlinha?>" class="TituloTd"><a href="#"
onclick="JanelaCentralizada('cadastro_qos.php?p=<?echo $linha['PROTO'];?>',300,200);"
STYLE="color:#000000;text-decoration: none;"><font>
  <? echo  $linha['PRIORIDADE'] ?></a></font></td>
  <td width="3%" bgcolor="<?echo $corlinha?>" class="TituloTd" title="vizualizar grafico rrd"><a href="#"
onclick="JanelaCentralizada('grafico.php?proto=<?echo $linhap."&sentido=".$sentido;?>',600,270);"
STYLE="color:#000000;text-decoration: none;"><font>
  <img src="img/bGraphIco.gif" border=0></td>
  </tr>
  <?
          $Resultado->MoveNext();
          }
          ?></table><br><?
    }
}
?>
```

Script rrdgraph.sh

```
#!/opt/bin/bash
prefix="/opt/share/www/top8/";
rrd="/opt/bin/rrdtool";
dir=$prefix$2;
l=`echo $1| sed s/-/_/g`;
#echo -n `date` atualizando o grafico para $l ...
$rrd graph $dir/$l.png -a PNG -t "Top 8 - Aplicacao ($l)" --vertical-label 'bits por segundo' \
      DEF:$l=$dir/$l.rrd:$l:AVERAGE \
      VDEF:"$l"max="$l",MAXIMUM \
      VDEF:"$l"avg="$l",AVERAGE \
      "CDEF:"$l"bits="$l",8,*" \
      --start end-20min \
      AREA:"$l"#FF0000:"$l" \
      COMMENT:"Maxima " \
      GPRINT:"$l"max:"%6.2lf %Sbps" \
      COMMENT:"Media " \
      GPRINT:"$l"avg:"%6.2lf %Sbps\l"
```