MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE MESTRADO EM SISTEMAS E COMPUTAÇÃO

JERONYMO MOTA ALVES DE CARVALHO

ARQUITETURA PARA CONTROLE DE CONGESTIONAMENTO E TARIFAÇÃO DE TRÁFEGO NÃO-COOPERATIVO

Rio de Janeiro
2009

INSTITUTO MILITAR DE ENGENHARIA

JERONYMO MOTA ALVES DE CARVALHO

ARQUITETURA PARA CONTROLE DE CONGESTIONAMENTO E TARIFAÇÃO DE TRÁFEGO NÃO-COOPERATIVO

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para obtenção do título de Mestre em Sistemas e Computação.

Orientador: Prof. Ronaldo M. Salles - Ph.D.

Rio de Janeiro
2009

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80-Praia Vermelha
Rio de Janeiro-RJ CEP 22290-270

C331a Carvalho, J. M. A.
Arquitetura para Controle de Congestionamento e Tarifação de Tráfego Não-Cooperativo/ Jeronymo Mota Alves de Carvalho. – Rio de Janeiro: Instituto Militar de Engenharia, 2009.
73 p.: il.

Dissertação (mestrado) – Instituto Militar de Engenharia – Rio de Janeiro, 2009.

1. Redes de computadores. 2. Tráfego não-cooperativo. 3. Tarifação. I. Título. II. Instituto Militar de Engenharia.

CDD 629.04

INSTITUTO MILITAR DE ENGENHARIA

JERONYMO MOTA ALVES DE CARVALHO

ARQUITETURA PARA CONTROLE DE CONGESTIONAMENTO E TARIFAÇÃO DE TRÁFEGO NÃO-COOPERATIVO

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para obtenção do título de Mestre em Sistemas e Computação.

Orientador: Prof. Ronaldo M. Salles - Ph.D.

Aprovada em 26 de janeiro de 2009 pela seguinte Banca Examinadora:

Prof. Ronaldo M. Salles - Ph.D. do IME - Presidente

Prof. Magnos Martinello - Ph.D. da UFES

Prof. Sidney Cunha de Lucena - DSc. da UNIRIO

Prof. Raquel Coelho Gomes Pinto - DSc. do IME

Rio de Janeiro
2009

Dedico esta dissertação à minha esposa Emanuelle e aos meus pais José e Marilena.

AGRADECIMENTOS

Primeiramente à Deus por sempre estar junto, principalmente nos momentos mais difíceis, dando força e serenidade para seguir em frente.

Ao meu orientador, Ronaldo M. Salles, pelo incentivo, disponibilidade e críticas fundamentais para realização deste trabalho.

À minha esposa Emanuelle, pelo apoio e compreensão de meus momentos de ausência dedicados à esta dissertação.

Por fim, à todos os professores e funcionários do Departamento de Engenharia de Sistemas (SE/8) do Instituto Militar de Engenharia.

Jeronymo Mota Alves de Carvalho

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ABREVIATURAS

AIMD	-	*Additive Increase/ Multiplicative Decrease*
CQ	-	*Custom Queueing*
DCCP	-	*Datagram Congestion Control Protocol*
ECSFQ	-	*Enhanced Core-Stateless Fair Queueing*
FIFO	-	*First-in-first-out*
IETF	-	*Internet Engineering Task Force*
IME	-	*Instituto Militar de Engenharia*
IP	-	*Internet Protocol*
MIB	-	*Management Information Base*
MSS	-	*Maximum Segment Size*
NBP	-	*Network Border Patrol*
OM	-	*Organização Militar*
PQ	-	*Priority Queueing*
QoS	-	*Quality of Service*
RED	-	*Random Early Detection*
RoI	-	*Return On Investment*
RTT	-	*Round-trip Time*
SCTP	-	*Stream Control Transmission Protocol*
SNMP	-	*Simple Network Management Protocol*
SRED	-	*Stabilized Random Early Detection*
TCP	-	*Transmission Control Protocol*
UDP	-	*User Datagram Protocol*
VoIP	-	*Voice over Internet Protocol*
WFQ	-	*Weighted Fair Queueing*

RESUMO

Este trabalho propõe uma arquitetura capaz de evitar o congestionamento do tráfego TCP causado pela intensa utilização de tráfego UDP, desestimulando o ingresso deste último sempre que o primeiro estiver experimentando baixo desempenho. Um esquema de tarifação e reserva de recursos é imposto a todo tráfego que utilize o protocolo UDP. O cálculo da tarifa é feito em função do prejuízo causado ao TCP pelo UDP. O conhecimento do estado dos enlaces é obtido através da utilização de tecnologia de fluxos. A reserva de recursos é feita através de regras de *firewall* e utilização do escalonador *custom-queueing*. Uma implementação da arquitetura foi realizada e testes comprovam a eficácia da solução.

ABSTRACT

This paper proposes an architecture capable of preventing TCP traffic congestion caused by the intense use of UDP traffic. It blocks new UDP connections when TCP traffic is experiencing low performance. A charging scheme and resource reservation procedure is imposed to all traffic that uses the UDP protocol. Prices are calculated according to the degre of starvation caused by UDP traffic over TCP flows. Knowledge about the status of flows is obtained through the use of flow protocol technology. Resources are reserved using firewall rules and custom-queueing mechanisms. An implementation of the architecture was made and tests showed the effectiveness of the solution.

1 INTRODUÇÃO

A Internet, originalmente concebida para uso de um pequeno grupo de pessoas, hoje é utilizada por aproximadamente 1,4 bilhão de pessoas em escopo mundial (STATS, 2008) (NETRATINGS, 2008). Mais do que uma rede bem sucedida, ela se faz presente e necessária em diversos aspectos da vida moderna, como pesquisa e estudo, relacionamento pessoal, economia e finanças, atividade profissional e entretenimento.

Com a convergência das redes (telefônica, rádio, televisiva, etc.), aplicações de VoIP (*voice over IP*) e *streaming* de áudio e vídeo tornaram-se extremamente comuns. Em julho de 2008, apenas nos Estados Unidos da América, foram assistidos 11,4 bilhões de vídeos online totalizando 558 milhões de horas (COMSCORE, 2008).

O protocolo de transporte TCP, largamente utilizado, possui uma filosofia de funcionamento diferente dessas aplicações. A principal crítica feita ao TCP é com relação ao *overhead* gerado desnecessariamente e ao baixo desempenho em aplicações multimídia. Por esses motivos, muitos programadores passaram a utilizar o protocolo de transporte UDP. Por não ser orientado à conexão, não implementar controle de fluxo ou congestionamento, por exemplo, ele permite um melhor desempenho.

Porém, a ausência de um mecanismo de controle de admissão ou de congestionamento no UDP pode provocar, por vezes, o estrangulamento do tráfego TCP de modo injusto. Ao perceber uma situação de congestionamento, o protocolo TCP reduz a taxa de envio de pacotes, diminuindo a sua vazão. Já o protocolo UDP continua enviando dados sem reduzir sua taxa de envio, forçando a diminuição da taxa de envio dos fluxos TCP.

Portanto, faz-se necessário algum tipo de mecanismo que evite ou pelo menos minimize os danos causados nesse cenário.

Uma possibilidade é tarifar o tráfego UDP de modo diferenciado dos demais visando desestimular seu uso irresponsável. A proposta é que o tráfego que prejudique os demais seja, de alguma forma, penalizado através de limitação de banda e de cobrança de créditos, sem deixar de prestar garantias a este tipo de tráfego para que o mesmo não seja banido da rede.

O presente trabalho propõe uma arquitetura capaz de evitar o congestionamento do tráfego TCP causado pelo UDP e, ao mesmo tempo, gerar garantias de qualidade para

este último, tarifando-o de acordo com o prejuízo causado às taxas de transmissão dos fluxos TCP. Desta forma, a arquitetura proposta desestimula o ingresso de fluxos UDP durante a ocorrência de congestionamento através de altas tarifas. Uma implementação da arquitetura proposta foi desenvolvida e experimentos práticos realizados em laboratório confirmaram a sua eficácia.

1.1 MOTIVAÇÃO

O problema do tema aqui abordado teve seus efeitos mais evidenciados após o surgimento dos enlaces de alta velocidade. Isto porque a utilização de aplicações multimídia com interatividade passou a ser viável em vários tipos de rede, como a Internet (SHENKER, 1995).

Como ainda não existe uma solução única que resolva integralmente o problema do comportamento não amigável do protocolo UDP com o TCP e que as velocidades de enlace tendem sempre à aumentar, é perfeitamente plausível imaginar um cenário futuro de congestionamento do tráfego TCP e/ou limitação severa do tráfego UDP (FLOYD, 1999).

A motivação do presente estudo é propor um mecanismo pelo qual o UDP não seja banido da rede mas que também o TCP não seja prejudicado sem que nenhuma compensação seja imposta ao tráfego não-cooperativo.

No âmbito do Exército Brasileiro, a presente proposta pode ser aplicada nas redes de dados que já estão em produção, como por exemplo a EBNet. Treinamentos e adestramentos realizados à distância através de video-aulas teriam garantia de performance ao mesmo tempo que o impacto deste tráfego na rede seria melhor controlado.

Aplicações de video conferência e até mesmo o Projeto C2 em Combate (COTER, 2002) também seriam beneficiados pela presente proposta, em especial este último, que possui diversos enlaces estreitos onde o problema de congestionamento é bastante comum.

Uma possibilidade é dotar o Módulo de Telemática do Exército com um sistema de créditos para cada Organização Militar que desejar utilizar tráfego UDP. Para transmitir fluxos UDP, a OM deve ter créditos suficientes para reservar o recurso necessário, caso contrário ela é impedida de transmitir a informação através desse protocolo. As comunicações seriam organizadas e garantias de qualidade oferecidas.

1.2 OBJETIVOS

Os principais objetivos deste trabalho são:

- desenvolver uma arquitetura capaz de evitar o congestionamento do tráfego TCP causado pelo UDP e, ao mesmo tempo, gerar garantias de qualidade para este último;
- tarifar as garantias prestadas aos fluxos UDP de acordo com o prejuízo causado às taxas de transmissão dos fluxos TCP;
- desestimular o ingresso de fluxos UDP durante a ocorrência de congestionamento através de altas tarifas;
- implementar a arquitetura proposta;
- validar a arquitetura proposta através de experimentos em ambiente real utilizando a implementação desenvolvida.

1.3 ORGANIZAÇÃO DA DISSERTAÇÃO

O capítulo 2 apresenta as principais características dos protocolos de transporte TCP e UDP, explicando o problema da coexistência destes protocolos no mesmo ambiente de rede, e discute as principais propostas de solução existentes que buscam sanar o problema.

No capítulo 3 a ocorrência do problema é evidenciada através de um experimento realizado no Laboratório de Redes da Seção de Engenharia de Computação do IME.

O capítulo 4 explica o funcionamento da arquitetura, justificando a escolha dos métodos de tarifação, escalonador e tecnologia para conhecimento do estado dos enlaces.

O capítulo 5 apresenta uma implementação da arquitetura, explicando suas especificidades.

No capítulo 6, experimentos são realizados com a finalidade de verificar a eficácia da arquitetura.

Por fim, o trabalho é concluído no capítulo 7, onde os resultados, objetivos alcançados, limitações encontradas e direções futuras são comentados.

2 UDP: UM PROTOCOLO NÃO AMIGÁVEL COM O TCP

Este capítulo apresenta as características dos dois principais protocolos de transporte utilizados atualmente, o TCP e o UDP, evidenciando a presença de controle de congestionamento no primeiro e sua ausência no segundo. Este comportamento não-cooperativo do UDP é designado por diversos autores como "*not TCP-friendly*" e pode gerar o grave problema conhecido como "*best-effort starvation problem*" descrito em (FLOYD, 1999).

Como o protocolo UDP não possui um mecanismo de controle de congestionamento, ele pode provocar o esgotamento dos recursos da rede ao enviar mais dados do que esta é capaz de trafegar. Os fluxos TCP concorrentes nessa rede percebem tal situação e diminuem suas respectivas taxas de envio de dados. Já os fluxos UDP permanecem enviando dados sem alterar seu comportamento e cada vez mais prejudicando os fluxos TCP.

Soluções para este problema já foram propostas por outros autores, mas, antes de discutí-las, uma breve revisão teórica sobre protocolos de transporte é necessária.

2.1 CONSIDERAÇÕES SOBRE PROTOCOLOS DE TRANSPORTE

A função deste tipo de protocolo é prover um mecanismo fim-a-fim para troca de dados entre sistemas finais.

Quando orientado à conexão, segundo (STALLINGS, 2000), sete são as questões que o protocolo de transporte deve tratar:

- Entrega ordenada
- Estratégia de retransmissão
- Detecção de dados duplicados
- Controle de fluxo
- Estabelecimento de conexão
- Término de conexão

- Recuperação a falhas

Dentro da suíte TCP/IP, os protocolos mais utilizados são o TCP (Protocolo de Controle de Transmissão), que é orientado à conexão, e o UDP (Protocolo do Datagrama do Usuário), que não é orientado à conexão. Outros protocolos menos populares são, por exemplo, o SCTP (Protocolo de Controle de Transmissão de *Stream*) e o DCCP (Protocolo de Controle de Congestionamento de Datagrama).

2.2 PROTOCOLO DE CONTROLE DE TRANSMISSÃO

Como já mencionado, o TCP é um protocolo orientado à conexão, trazendo confiabilidade às comunicações fim-a-fim mesmo em redes onde o meio de transmissão não seja confiável.

Seu cabeçalho possui 20 bytes de tamanho e uma conexão é identificada através da tupla IP de origem, porta de origem, IP de destino, porta de destino.

Mesmo depois da padronização do TCP pela RFC 793, diversas técnicas de controle de congestionamento surgiram, o que deu origem a RFC 2001. Foram criados mecanismos para evitar situações de congestionamento provocadas por ele próprio e, mesmo que esta ocorra devido à fatores externos, alterar seu comportamento procurando sanar o problema e se recuperar.

Atualmente, existem diversas variantes do TCP como Tahoe, Reno, New Reno, Sack, Westwood, BIC, CUBIC, HSTCP e Libra, dentre outros, segundo (KUROSE, 2004) e (MICHEL, 2007).

(LEE, 2008) explica que o TCP Reno, um dos esquemas reativos de controle de congestionamento mais largamente adotados em redes reais, possui quatro fases de transmissão: *slow start*, *congestion avoidance*, f*ast recovery* e *fast retransmit*. Este protocolo mantém duas variáveis: *cwnd* e *ssthresh*. No início de um congestionamento TCP, o emissor entra na fase *slow start*, na qual a *cwnd* é acrescida de uma unidade a cada ACK recebido. Em seguida, a variável *cwnd* do emissor cresce exponencialmente de acordo com o RTT. Quando *cwnd* alcança *ssthresh*, o emissor entra na fase *congestion avoidance*, na qual *cwnd* é aumentada linearmente por um ao receber o grupo de ACKs para os segmentos enviados. Quando uma perda de pacote ocorre devido ao congestionamento da rede, ou o emissor recebe DUPACKs (ACKs duplicados) ou o seu temporizador RTO expira, estes eventos ativam as fases f*ast retransmit* e *fast recovery*, através da qual cada emissor reduz o tamanho de sua variável *cwnd* pela metade e a aumenta linearmente como na fase *con-*

gestion avoidance. Perceba que a variável *cwnd* é ajustada baseada em ACKs, DUPACKs e RTO.

O mecanismo descrito é costumamente referenciada pela sigla AIMD (*Additive Increase / Multiplicative Decrease*). Sempre que uma situação de possível congestionamento é percebida (perda de pacotes) o protocolo diminui exponencialmente sua taxa de transmissão e procura, em seguida, aumentá-la de forma linear.

2.3 PROTOCOLO DO DATAGRAMA DO USUÁRIO

O protocolo UDP, descrito na RFC 768, não é orientado à conexão, e sim à transmissão. Seu cabeçalho possui 8 bytes de tamanho e as sete questões observadas por (STALLINGS, 2000) não são implementadas neste protocolo uma vez que seu intuito é ser um protocolo enxuto, com pouco *overhead*, e desempenho máximo na troca de mensagens.

Essas características fazem do UDP um protocolo adequado para aplicações inelásticas, onde a tolerância a atrasos e suas variações é baixa. Um caso típico é o VoIP (voz sobre IP), onde a perda de alguns pacotes causa pouco impacto à compreensão geral da mensagem.

Outro papel importante desempenhado pelo UDP é na gerência de redes. O protocolo SNMP (*Simple Network Management Protocol*) o utiliza para a troca de mensagens (MAURO, 2001).

Porém o uso excessivo deste protocolo pode provocar o congestionamento na rede conforme observado por (FLOYD, 1999).

2.4 DISCUSSÃO DAS SOLUÇÕES EXISTENTES PARA O PROBLEMA

Entre os artigos estudados de maior relevância, pode-se observar que as áreas de estudo e as soluções propostas podem ser agrupadas em três grandes áreas, a saber:

- Soluções baseadas nos *Hosts*
- Soluções baseadas nas Redes
- Soluções baseadas em Preço

As soluções baseadas nos *Hosts* são aquelas onde seria necessário efetuar mudanças em cada *host* da Internet, bem como em todos os ativos de rede. Possui a desvantagem de ser necessário alterar cada *host* em toda a rede.

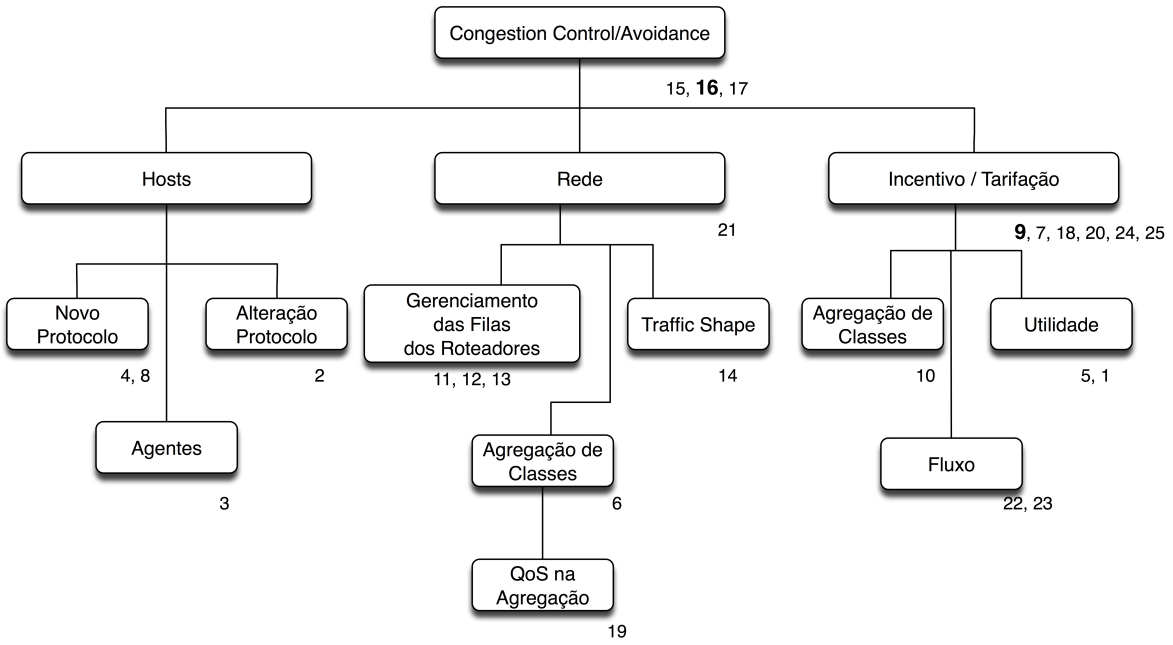

FIG. 2.1: Mapeamento dos artigos mais relevantes estudados

Soluções baseadas na Rede são aquelas onde os roteadores da rede devem ser alterados para que o tráfego UDP não sufoque o tráfego TCP. São mais vantajosas em relação às soluções baseadas nos *Hosts*, pois a quantidade de ativos a serem alterados é bem menor, porém, em geral, ocorre algum tipo de punição grave, como descarte de pacotes ou limitação de banda, ao tráfego UDP.

Já as soluções baseadas em Preço, tal qual a anterior, necessitam alterar uma quantidade razoável de ativos na rede, porém não punem severamente o UDP e sim possuem tarifas maiores pelo serviço prestado. O valor arrecadado pode ser usado para ampliar/melhorar a rede ao mesmo tempo que desestimula o tráfego UDP menos importante.

Para que se possa melhor compreender o universo de soluções propostas, a FIG. 2.1 apresenta um mapeamento dos artigos mais importantes estudados e a TAB. 2.1 possui as respectivas referências.

O artigo mais antigo estudado durante a fase de pesquisas foi o de John Nagle. Nele, o autor observa o fato do protocolo TCP, na época, não possuir mecanismos que resolvessem de maneira inequívoca situações de congestionamento na rede (NAGLE, 1984).

Essas observações estimularam a criação de mecanismos eficazes de controle e prevenção de congestionamento. Surge, a partir desses mecanismos, o conceito de que os

TAB. 2.1: Legenda da FIG. 2.1

Item	Referencia
1	(RIBEIRO, 2007)
2	(LAI, 2004)
3	(VAN, 2006)
4	(KOHLER, 2006)
5	(MANI, 2006)
6	(YILMAZ, 2001)
7	(FLOYD, 1999)
8	(CHODOREK, 2000)
9	(COCCHI, 1993)
10	(WANG, 2006)
11	(WANG, 2003)
12	(TANG, 2004)
13	(GAO, 2005)
14	(ALBUQUERQUE, 2004)
15	(NAGLE, 1984)
16	(SHENKER, 1995)
17	(LEFELHOCZ, 1996)
18	(SHENKER, 1996)
19	(XU, 2005)
20	(BYUN, 2004)
21	(OTT, 1999)
22	(ROBERTS, 2004)
23	(KEY, 2004)
24	(MACKIE-MASON, 1995)
25	(BRISCOE, 2007)

protocolos devem ser amigáveis com o TCP (*TCP friendly*).

Apesar do exposto, é imperioso notar que a primeira referência sobre o conflito específico do UDP com o TCP foi apresentada por Scott Shenker *et al* (LEFELHOCZ, 1996), porém a idéia de apreçamento em rede de computadores pertence a outro trabalho de Scott Shenker *et al* (COCCHI, 1993).

Shenker explica as necessidades e as diferenças entre aplicações elásticas e as de tempo real. As elásticas toleram atrasos e perda de pacotes sem prejuízo da aplicação, enquanto as de tempo real são sensíveis a esses tipos de problema (SHENKER, 1995). Por isso mesmo, as aplicações de tempo real são desenvolvidas para extrair o máximo da rede, mesmo que atrapalhem aplicações elásticas.

Enquanto o TCP faz uso do mecanismo AIMD, o UDP não o faz. Este mecanismo faz com que a taxa de envio de pacotes de um fluxo TCP aumente linearmente enquanto nenhum evento de congestionamento seja percebido, e diminua exponencialmente quando o for. A ausência deste mecanismo no protocolo UDP permite que ele obtenha um *throughput* muito maior do que o de um fluxo TCP concorrente e, por isso, não seja cooperativo com o TCP (LEFELHOCZ, 1996).

Em (LAI, 2004) demonstra-se que as características do TCP de tráfego em rajada e flutuação do tamanho da janela de transmissão impossibilitam a sua utilização para aplicações de tempo real, preferindo estas utilizar o UDP.

Ressalta-se que a demanda por aplicações multimídia vem aumentando nos últimos anos e já representa cerca de um quinto do tráfego da Internet e, por isso, aproximadamente 10% do tráfego da Internet seja devido ao protocolo UDP (KEY, 2004).

Para (VAN, 2006), uma solução seria a adoção de um *framework* baseado em agentes. Sempre que um tráfego irresponsável fosse identificado, agentes seriam instanciados para atuar sobre a transmissão e torná-la amigável com o TCP. Tal solução, porém, gera problemas de escalabilidade e *overhead* na rede.

Já para (LAI, 2004), uma alternativa ao uso do UDP é alterar o TCP de maneira a incorporar suavidade nas taxas de transmissão. Essa proposta se faz presente através do algoritmo de controle de congestionamento Slack Term. Tal alternativa, porém, é pouco viável do ponto de vista prático uma vez que seria necessário alterar o protocolo TCP utilizado em todos os *hosts* da Internet.

Uma outra alternativa é apresentada por Sally Floyd *et al* (KOHLER, 2006). Ao invés de se utilizar agentes ou *patches* no TCP, um novo protocolo de transporte deve ser

acrescentado ao mundo TCP/UDP: o DCCP.

O DCCP, Datagram Congestion Control Protocol, é baseado no UDP, sendo quase tão leve quanto ele, porém implementa alguns mecanismos de controle de congestionamento, como a transmissão de informações de *acknowledgement*. Infelizmente, esse protocolo ainda não é um produto acabado e esta solução possui a desvantagem de se ter que alterar o protocolo de transporte utilizado por todas as aplicações existentes.

Com relação a mecanismos de controle de congestionamento, Scott Shenker ressalta que cada aplicação possui uma função utilidade diferente e propõe que controles de admissão na rede devem ser desenvolvidos para evitar congestionamento baseado nas mesmas (SHENKER, 1995). Ele ainda analisa os algoritmos conhecidos como RED e DECbit.

O RED (*Random Early Detection*) consiste, basicamente, em descartar pacotes com uma probabilidade dependente da carga sempre que o *buffer* do roteador aparentar estar congestionado.

Uma evolução do mecanismo acima citado é o SRED (*Stabilized RED*). Ele consegue estabilizar a ocupação do buffer do roteador independente do numero de conexões ativas (OTT, 1999).

Uma grande contribuição do SRED é que seu mecanismo de preenchimento da lista de zumbis permite estimar estatisticamente o número de conexões ativas e encontrar fluxos "mal comportados" que estejam transmitindo mais que os outros.

Outra maneira de proteger o tráfego TCP do UDP é alterar o gerenciamento de filas nos roteadores. Um bom exemplo é o CHOKe. Jiantao Wang *et al* explicam o funcionamento deste algoritmo e demonstram, através de um modelo analítico, que o tráfego TCP é preservado quando existe um único fluxo UDP concorrente (WANG, 2003)(TANG, 2004). A maior crítica a este esquema é justamente o fato dele não funcionar satisfatoriamente na presença de mais de um único fluxo UDP e não prever topologias com mais de uma rota.

Uma outra abordagem com relação ao gerenciamento das filas dos roteadores é dada por (GAO, 2005). Utilizando os conceitos de teoria dos jogos, eles propõem um jogo onde cada fluxo é um jogador que procura aumentar sua utilidade (*throughput*) sem negociar com os demais, não ocorrendo nenhum tipo de compensação e tendo sempre em vista o equilíbrio de Nash. Um grande empecilho para utilização deste sistema é que ele foi concebido para operar com fluxos que chegam com taxas constantes nos roteadores, sendo este um caso pouco realístico.

Ainda com relação ao gerenciamento de filas nos roteadores, (ALBUQUERQUE, 2004) propõem o mecanismo de prevenção contra congestionamento NBP (*Network Border Patrol*). Como o nome já sugere, ele atua apenas nos roteadores de borda das redes, não sendo necessário alterar todos os roteadores da Internet.

Sempre que um fluxo irresponsável é detectado no roteador de ingresso da rede, políticas de restrição são adotadas para este fluxo. Mais do que isso, quando o NBP é utilizado junto com o enfileiramento no roteador do tipo ECSFQ (*Enhanced Core-Stateless Fair Queueing*), a alocação de banda é feita de maneira justa entre os fluxos competidores.

A fragilidade desta proposta é a sua escalabilidade. Como ela monitora o estado de cada fluxo nos roteadores de ingresso, sempre que um roteador de borda estiver conectado a muitos outros, o número de fluxos ativos for grande ou a carga de tráfego for pesada, esta proposta se torna inviável.

O problema do controle de congestionamento do TCP ser vulnerável em ambientes onde fluxos com sensibilidades de congestionamento diferentes, como é o caso do UDP, disputam recursos escassos, pode ser encarado através da utilização de classes de serviços. Este é o caso de (YILMAZ, 2001).

A arquitetura proposta é do tipo DiffServ com utilização de três classes. Cada roteador trabalharia com um fila para o tráfego UDP, outra para o tráfego TCP longo e mais outra para o tráfego TCP curto. Através do isolamento destes protocolos de transporte em classes distintas, espera-se obter um comportamento mais justo nas transmissões, bem como a possibilidade de controlar parâmetros de QoS para determinados fluxos.

A QoS mencionada está relacionado ao tráfego agregado. O estudo conduzido em (XU, 2005) desenvolveu modelos analíticos que permitem calcular a probabilidade de perda de fluxos individuais em ambientes onde apenas a probabilidade de perda agregada pode ser controlada. Desta maneira, os autores observaram em seus experimentos que, para que a garantia dada ao tráfego agregado valha para os fluxos individualmente, alguns requisitos devem ser satisfeitos. Segundo os autores, quando o número de fluxos a serem agregados for pequeno, deve-se evitar fluxos com grande diferença de taxa de transmissão. Já quando o número de fluxos agregados for grande, deve-se evitar fluxos que possuam a sua taxa média de transmissão muito menor que a taxa de pico.

Além das propostas já apresentadas, este estudo também analisou aquelas que adotam esquemas de tarifação de tráfego. Tais esquemas cobram tarifas extras pela utilização dos recursos de rede em situações específicas tendo como base acordos previamente estabele-

cidos.

Em geral, a cobrança ocorre quando se deseja alguma garantia extra sobre os recursos disponíveis ou quando ocorre falta de recursos e alguma prioridade deve ser adotada.

Pode-se encarar a falta de recursos na rede como uma má gerência do administrador. Em sistemas comerciais, a equipe responsável pela gerência de tráfego costuma aumentar a capacidade dos enlaces antes que a escassez de recursos seja atingida.

Esse tipo de visão faz com que o usuário contratante do serviço não esteja disposto a pagar pelo erro do contratado. Por isso, esquemas de apreçamento em situações de congestionamento, como o *smart market* (MACKIE-MASON, 1995), estão fadados ao insucesso. Alguma garantia da qualidade do serviço deve ser dada ao cliente.

Nesse sentido, o trabalho de Shenker *et al* é mais aplicável à realidade existente. Estes autores sugerem a classificação do tráfego em classes de serviço de acordo com as garantias de qualidade de serviço oferecidas. Quanto maior for a qualidade, maior o preço cobrado. Os usuários podem, desta maneira, escolher a tarifa cobrada de acordo com a classe que desejam ocupar (SHENKER, 1996).

Sally Floyd aprova o uso de esquemas de tarifação uma vez que estes servem para desestimular o uso de tráfego irresponsável (não amigável com TCP). Na verdade, sua sugestão completa é composta por mais duas idéias. A segunda é a utilização de políticas de agendamento nas filas dos roteadores que isolem os fluxos, tanto quanto possível, da influência de outros fluxos. A terceira é a utilização de controle de congestionamento nos roteadores de forma que, sempre que situações de congestionamento sejam atingidas, o próprio roteador restrinja a largura de banda de tráfego do tipo melhor-esforço de modo a estimular nos usuários e desenvolvedores a utilização de protocolos que realizem o controle fim-a-fim de fluxos deste tipo.

Para que o esquema proposto pela autora funcione é necessário que se identifique os fluxos irresponsáveis. A proposta é identificar os fluxos que estejam utilizando muito mais banda do enlace que os demais em situações de congestionamento e/ou que possuam taxa de chegada maior que um valor determinado por uma equação baseada no tamanho máximo do pacote, o atraso de propagação do enlace considerado e a taxa de descarte de pacotes na fila do roteador.

Porém, a própria autora admite a alta complexidade da proposta e conclui que o importante é que algum mecanismo que evite congestionamento causado por protocolos irresponsáveis seja desenvolvido (FLOYD, 1999).

Em (ROBERTS, 2004) afirma-se que, para que um sistema de tarifação em rede de computadores obtenha êxito, é necessário que se obedeça aos quatro princípios descritos a seguir:

- o retorno do investimento (RoI, da sigla em inglês) seja assegurado pela divisão do capital da rede e dos custos de operação entre os usuários;
- a discretização do preço, apesar de economicamente eficiente, deve ser baseado em outros critérios além da garantia da qualidade do serviço prestado;
- a cobrança tão somente em situações de congestionamento, usado com eficiência para dividir os recursos já escassos, não é satisfatório como base de cobrança em redes comerciais;
- a preferência por simplicidade e transparência pode ser satisfeita por um esquema de cobrança que utilize controles de admissão.

Para Byun e Chatterjee, além dos princípios já descritos acima, a busca de mecanismos baseados em preço que evitem o congestionamento da rede deve incluir o princípio da maximização do lucro (BYUN, 2004).

Os autores citam diversos trabalhos que mostram que, de maneira geral, clientes preferem ter certeza ou pelo menos uma precisa previsão de seus gastos por um determinado produto adquirido ou serviço prestado, mas eles aceitam pagar mais pela garantia da qualidade.

É por este motivo, segundo (WANG, 2006), que usuários leves (que não fazem uso de aplicações de tempo real e realizam pouco *download*) preferem assinar serviços como se fossem usuários pesados.

Esses autores destacam que a cobrança pela utilização da Internet é feita de forma fixa de acordo com a velocidade de acesso, o volume de dados transmitidos ou uma combinação de ambos.

A cobrança de preço fixo, apesar de largamente utilizada, não é eficiente e faz com que usuários leves subsidiem os usuários pesados, tornando-se um mecanismo injusto de cobrança.

Portanto, deve-se buscar mecanismos mais justos baseados em tarifação diferenciada conforme o perfil do usuário.

Tendo como variáveis o consumo de recursos da rede e a qualidade de serviço desejada, funções de investimento podem ser utilizadas para gerar o valor devido para determinado fluxo e ainda distribuir de maneira mais justa o largura de banda disponível através da suavização de diferenças de distribuições entre estes (MANI, 2006).

O estudo de Salles e Ribeiro simula um esquema de roteamento e controle de admissão baseado nas funções utilidade para tráfego TCP e UDP. A proposta é que o tráfego UDP seja tarifado de acordo com a desutilidade causada ao TCP (RIBEIRO, 2007).

As soluções do tipo *Host* ou Rede, isoladamente, de acordo com a FIG. 2.1, mostram-se incapazes de solucionar o problema de estrangulamento do TCP devido a dificuldades de escalabilidade e complexidade de implementação.

Outro fator que deve ser considerado nas propostas pesquisadas é: o que se busca alocar de modo justo? Segundo Bob Briscoe, grande parte dos estudos conduzidos nos últimos dez anos fracassaram ou ainda vão fracassar porque abordam o problema com uma visão errada (BRISCOE, 2007). Para o autor, não se deve buscar simplesmente alocar os fluxos e suas taxas de transmissão de modo a alcançar um ambiente mais justo, mas sim alterá-los de modo que os custos de transmissão sejam rateados de modo justo.

A proposta desta dissertação é utilizar partes dos mecanismos das soluções baseadas em rede e em preço para criar uma arquitetura híbrida que seja melhor do que as soluções apresentadas em separado. Espera-se, desta forma, conseguir controlar o congestionamento de tráfego que ocorra devido ao problema da coexistência de fluxos UDP e TCP nas redes de computadores. Mais do que isso, a arquitetura aqui proposta garante uma QoS mínima para o tráfego TCP e disponibiliza um sistema de compra de QoS para os usuários de tráfego UDP, além de permitir ao administrador da rede fixar um percentual máximo de utilização dos enlaces para tráfego UDP baseado nas tarifas cobradas.

3 O PROBLEMA

Uma rede de dados que não posssua políticas de controle de tráfego está sujeita a ter seu tráfego TCP estrangulado sempre que se fizer uso intenso do protocolo UDP. Isto ocorre porque os fluxos UDP são irresponsáveis e não amigáveis com os fluxos TCP.

Para obter maior desempenho, o protocolo UDP não realiza controle de congestionamento, simplesmente envia os dados. Já o protocolo TCP implementa mecanismos para evitar o congestionamento e se recuperar sempre que o mesmo ocorra, evitando o colapso da rede.

Devido a esta característica, sempre que uma rede de dados está sendo utilizada em sua capacidade máxima ou próxima desta, o protocolo TCP diminui exponencialmente a taxa de envio de seus pacotes para evitar o colapso da rede. Quando o protocolo percebe que não mais existe o risco do congestionamento, ele aumenta a taxa de envio de pacotes linearmente.

O problema se evidencia quando os fluxos UDP enviam uma quantidade de dados que exija um *throughput* total maior que a largura de banda disponível. Nesta situação, os fluxos TCP percebem a situação de congestionamento e diminuem as suas respectivas taxas de envio, podendo mesmo chegar a situação onde o tamanho da janela de envio de dados é reduzido ao tamanho unitário.

Esta situação de estrangulamento dos fluxos TCP é mantida enquanto não houver redução da taxa de envio dos fluxos UDP. Para comprovar o problema descrito, foram realizados quatro experimentos controlados no Laboratório de Redes da Seção de Engenharia de Computação do Instituto Militar de Engenharia. Os dois primeiros demonstram a capacidade dos protocolos TCP e UDP em utilizar plenamente a banda disponível nos enlaces. O terceiro experimento demonstra a capacidade do protocolo TCP em dividir equalitariamente a banda do enlace entre os seus fluxos ativos. Por fim, o quarto exprimento evidencia a ocorrência do problema descrito.

A obtenção e análise dos dados foi realizada através da ferramenta ManageEngine NetFlow Analizer (ADVENTNET), ressaltando que os roteadores do laboratório foram previamente configurados para exportar as informações de tráfego dos fluxos de dados através da funcionalidade NetFlow versão 5 presente nos respectivos sistemas operacionais.

A ferramenta utilizada é composta por dois módulos. O primeiro é um coletor de fluxos NetFlow que salva as informações provenientes dos roteadores em um arquivo que funciona como uma base de dados com formato proprietário. O segundo módulo é uma interface web capaz de executar buscas na base de dados e gerar relatórios para posterior análise.

O cenário utilizado está ilustrado na FIG. 3.1 através de um "corte" do esquema de rede do Laboratório. Os dois roteadores são Cisco. O modelo 1750 é identificado como cyprus e seu IOS é versão 12.1. Já o modelo 2611 é identificado como creta e seu IOS é versão 12.2. O enlace que conecta esses dois roteadores é serial e foi configurado para operar em 1 Mbit/s com MTU de 1500 bytes. A configuração da velocidade do enlace foi feita alterando o *clockrate* da porta do roteador configarada para operar como DCE (*Data Circuit-terminating Equipment*).

As duas estações de trabalho que participaram do teste pertencem a subredes diferentes. A primeira estação é identificada como grecia01, possui ip 192.168.0.1, está localizada na subrede 192.168.0.0 e está conectada ao roteador cyprus através de um enlace FastEthernet (100 Mbit/s). A segunda estação é identificada como grecia02, possui ip 192.168.1.1, está localizada na subrede 192.168.1.0 e está conectada ao roteador creta através de um enlace Ethernet (10 Mbit/s). Ambas utilizam o sistema operacional openSuse 11.0 e placas de rede Ethernet 10/100/1000 Mbps.

Para gerar o tráfego entre as estações de trabalho foi desenvolvido um gerador de tráfego do tipo CBR (*constant bitrate*). Como funcionalidades deste gerador é possível escolher o tamanho dos pacotes enviados, a frequência de envio e o tipo de protocolo de transporte utilizado (TCP ou UDP).

Os três primeiros testes tiveram duração de 60 s com pacotes de tamanho 1250 bytes emitidos a cada 10 ms pela estação grecia01 destinados à estação grecia02. No caso dos fluxos TCP, esse tempo varia conforme o congestionamento do enlace. As escolhas dos valores de frequência de emissão e tamanho de pacote foram feitas de modo a utilizar totalmente a capacidade pré-configurada do enlace serial, o qual apresentou, durante os testes, um retardo médio de 2 ms.

A FIG. 3.2 contêm os resultados obtidos.

Os testes 1 e 2 mostram que, em separado, os protocolos TCP e UDP são capazes de utilizar quase que a totalidade dos recursos disponíveis. O UDP utilizou 97,7% da capacidade do enlace serial entre os roteadores cyprus e creta. O TCP utilizou 96,6% da capacidade do mesmo enlace. Os 100% teóricos não foram atingidos devido ao tráfego

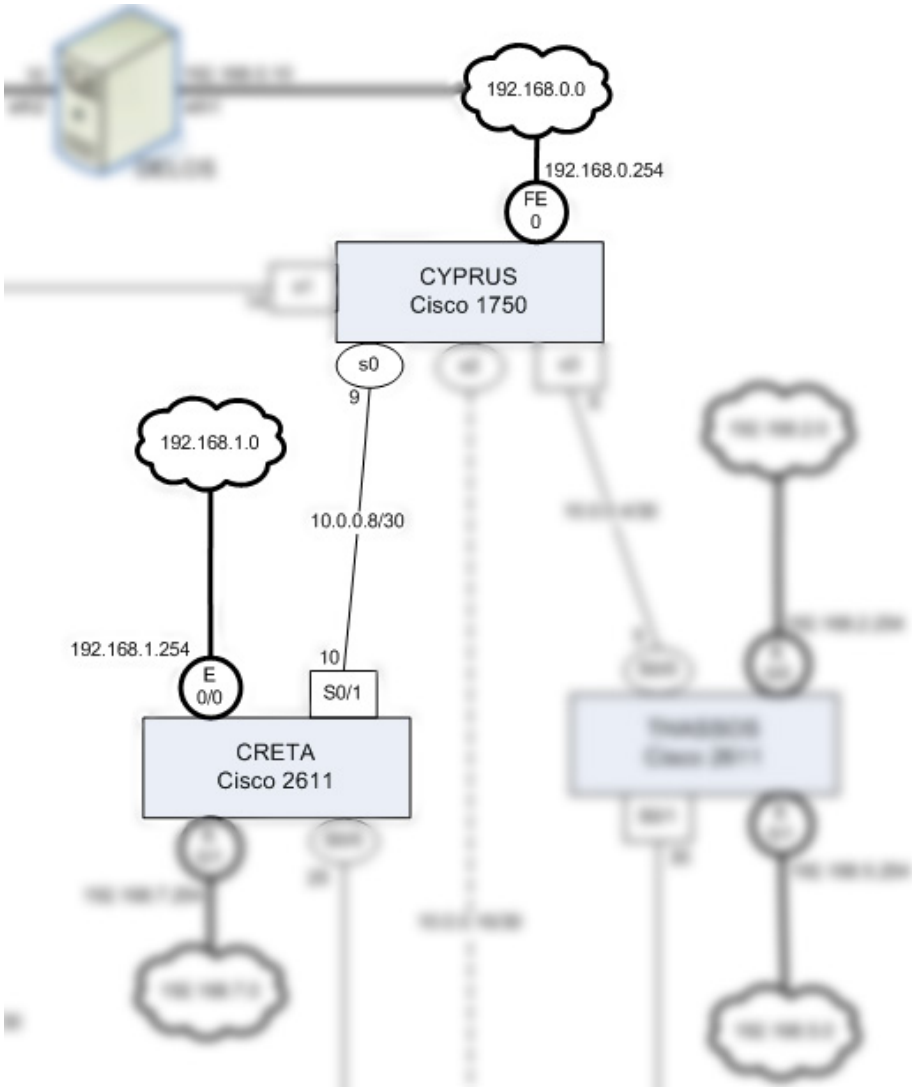

FIG. 3.1: Cenário dos Testes.

Teste	Protocolo	Fluxos	Enviado (bytes)	Recebido (bytes)	Aproveitamento (%)	Utilização	
						Por Fluxo	Agregada
1	UDP	1	7467500	7327500	98,13	97,70	97,70
2	TCP	1	7408750	7244638	97,78	96,60	96,60
3	TCP	3	2054032	1953922	95,13	26,05	95,95
			2753750	2650694	96,26	35,34	
			2713750	2591426	95,49	34,55	

FIG. 3.2: Resultados dos Testes 1, 2 e 3.

de mensagens de comunicação entre os roteadores e dos *exports* dos roteadores com as próprias informações de fluxos.

Já o teste 3 evidencia o comportamento amigável dos fluxos TCP. Três fluxos deste protocolo foram gerados entre as estações grecia01 e grecia02 e todos aproveitam o recurso disponível no enlace serial entre os roteadores de forma equilibrada em aproximadamente 30% cada. Foi observado que um dos fluxos TCP teve desempenho inferior aos demais. Acredita-se que tal fato pode estar relacionado à questões de implementação do sistema operacional utilizado, como tarefas concorrentes e escalonador de processos.

O teste 4, cujos resultados estão consolidados na FIG. 3.3, é uma variação do terceiro teste onde, além dos três fluxos TCP já mencionados, um quarto fluxo UDP foi acrescido entre as estações grecia01 e grecia02. O tempo de simulação foi aumentado totalizando 11 minutos. O experimento foi conduzido de tal forma que, durante o primeiro minuto de teste, não fosse gerado nenhum fluxo UDP e, a partir do segundo minuto e até o final, um único fluxo UDP concorrente aos fluxos TCP já existentes fosse gerado entre as duas estações de trabalho e a quantidade de dados emitidos pelo fluxo UDP fosse aumentada a cada minuto. A variação do *throughput* UDP foi obtida através da diminuição do período entre a geração de pacotes UDP. O experimento, a partir do segundo minuto, começa com um período de 100 ms entre a geração de pacotes UDP e vai diminuindo a cada minuto em 10 ms, terminando no 11º minuto com um período igual a 10 ms.

A linha azul representa o tráfego UDP gerado pela estação emissora enquanto que a linha vermelha representa o tráfego UDP que realmente atravessou o enlace serial e foi recebido pela estação receptora. Observa-se que o *throughput* UDP é sempre maior que *goodput* UDP, chegando a 100% da capacidade do enlace no final do teste. A diferença entre o *throughput* e o *goodput* representa os pacotes descartados na transmissão.

A linha verde representa o *goodput* agregado dos três fluxos TCP. Já a linha roxa representa o *goodput* agregado do enlace (os três fluxos TCP mais o fluxo UDP).

A FIG. 3.3 mostra que, ao final do teste 4, o fluxo UDP domina a utilização do enlace serial enquanto que os três fluxos TCP agregados praticamente não transmitem informação.

Fica demonstrado neste último teste a ausência de comportamento amigável de um fluxo UDP para com os fluxos TCP.

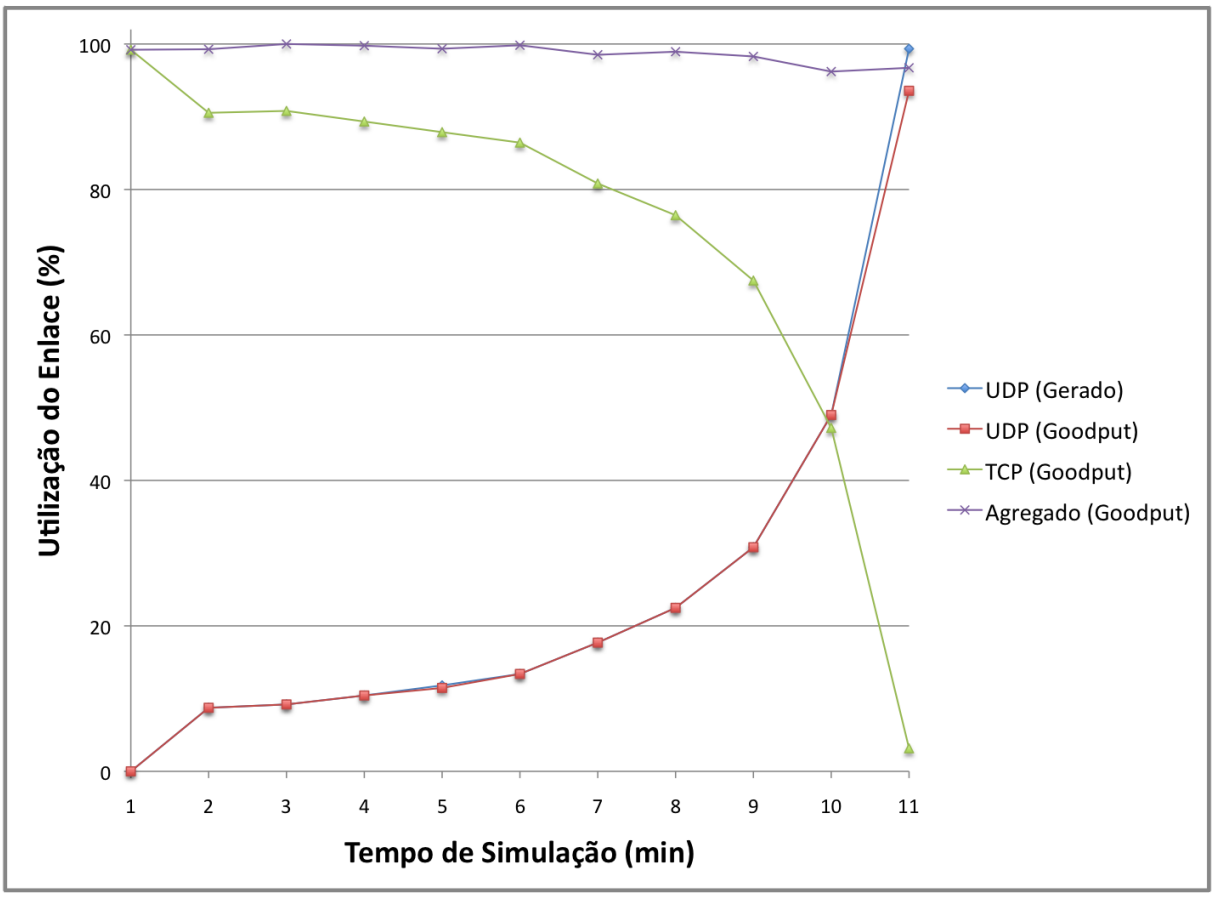

FIG. 3.3: Gráfico dos resultados obtidos no teste 4.

4 ARQUITETURA PROPOSTA

A solução aqui proposta é hibrida, pois utiliza tanto os conceitos das soluções baseadas em Rede quanto as soluções baseadas em Preço, e não é centralizada em torno de uma tecnologia específica.

O sistema foi concebido utilizando a figura de um gerente que monitora e realiza alterações na rede ao longo do tempo. Este gerente recebe a denominação de Bandwidth Broker.

O Bandwidth Broker deve ser alimentado com informações dos usuários e da rede para que possa reconfigurar esta última. A obtenção destas informações e o modo como é feita a reconfiguração constituem os diversos subsistemas da arquitetura, sendo o próprio Bandwidth Broker um deles. A FIG. 4.1 ilustra os diversos subsistemas.

4.1 BANDWIDTH BROKER

De acordo com (NICHOLS, 1999), Bandwidth Broker é um agente da rede configurado com políticas organizacionais, capaz de gerenciar a alocação de recursos de um tipo de tráfego específico e interpretar novos pedidos de alocação baseado nas políticas instituídas e no tráfego corrente.

Na arquitetura aqui proposta, o Bandwidth Broker deve se comunicar com os roteadores e *hosts* da rede sob sua responsabilidade para exercer as funções de gerência da rede de dados.

Antes que uma conexão UDP seja aceita, o Bandwidth Broker deve receber um pedido proveniente do usuário informando os dados pertinentes à transmissão. Após determinar a rota e o custo associado a esta, o Bandwidth Broker aloca os recursos necessários de forma a garantir os requisitos da transmissão.

A rota e o seu custo associado devem ser calculados de acordo com os conceitos de microeconomia e funções utilidade aplicados aos fluxos UDP e TCP.

O Bandwidth Broker deve ser informado sobre o perfil financeiro de cada usuário de modo que conexões sejam aceitas sempre que o perfil do usuário suportar o custo associado. Neste caso, o valor deve ser registrado para posterior cobrança. Se o perfil do usuário não suportar o custo associado, a transmissão deve ser bloqueada.

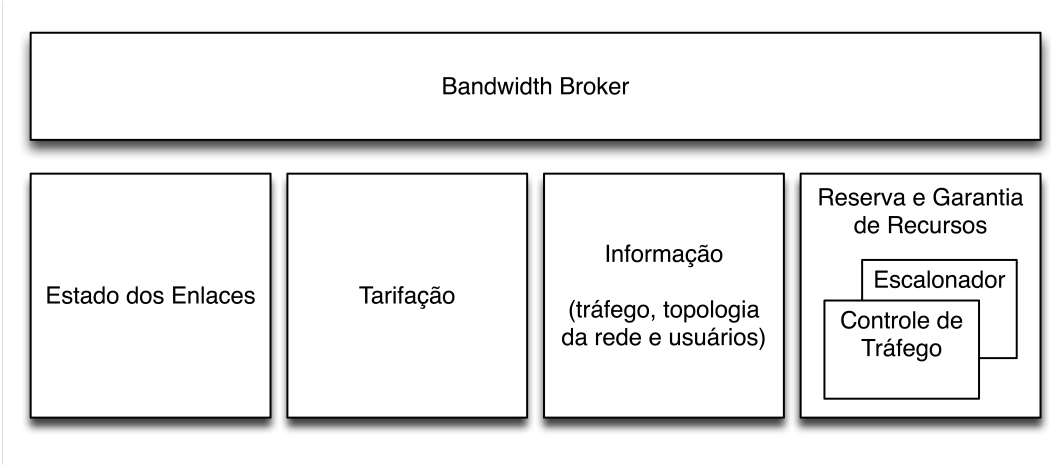

FIG. 4.1: Subsistemas da arquitetura proposta.

A garantia do serviço prestado ao tráfego UDP deve ser feita através do subsistema Reserva e Garantia de Recursos.

4.2 TARIFAÇÃO

O preço a ser cobrado de cada fonte de tráfego UDP depende se esta está contribuindo de modo injusto para a utilização do enlace por parte do tráfego TCP.

Função utilidade é um conceito da microeconomia que representa o prazer obtido (satisfação) com a variação da quantidade possuída de um determinado bem (STIGLER, 1970).

Já a satisfação é um conceito subjetivo, pois o que é suficiente para um usuário pode não ser para outro. Uma rede com todos os enlaces FastEthernet mas com atraso total entre as bordas de 300 ms é suficiente para a transferência de um arquivo, mas claramente insuficiente para uma aplicação de VoIP, onde a recomendação para interatividade é de atraso máximo de 150 ms segundo a norma ITU-T G.114.

Na área de redes de dados, a função utilidade pode ser utilizada para medir a satisfação de um usuário ao utilizar determinada quantidade de recursos disponíveis (RIBEIRO, 2007). O modo utilizado para quantizar a percepção negativa do tráfego TCP, e consequentemente tarifar o UDP, é feito através de funções de desutilidade.

O cálculo da desutilidade sentida pelo TCP é feito através da função proposta por (RICHARDS, 1999). Este estudo propõe um mecanismo de QoS através do controle de dois parâmetros: nível de satisfação do usuário com a QoS utilizada e o preço atrelado

ao mesmo. A arquitetura nele desenvolvida permite traduzir um parâmetro de qualidade da aplicação para uma componente de satisfação através de uma função logarítmica de apenas uma variável.

No caso da solução aqui desenvolvida foi escolhida como variável o *throughput* do tráfego TCP.

4.2.1 UTILIDADE TCP

A utilidade para os usuários que utilizam este protocolo em suas aplicações está relacionada ao desempenho obtido na transmissão da informação desejada. Ou seja, é a relação entre a quantidade de dados úteis enviados (*Goodput*) e o total de dados enviados (*Throughput*).

O estudo de (RIBEIRO, 2007) demonstra que a utilidade TCP pode ser obtida através de uma equação com três variáveis

$$U_{tcp} = 1 - \left(\frac{MSS}{RTT * Throughput} \right)^2 \tag{4.1}$$

Porém aferir essas três variáveis em um cenário real é uma tarefa complexa. Aproveitando a técnica desenvolvida em (RICHARDS, 1999) para aferir satisfação em função apenas de uma variável, o presente trabalho utilizou o *throughput* como variável para calcular a utilidade do TCP.

$$U_{tcp} = a * ln\left(b * Throughput - c\right) \text{ onde } a = \frac{1}{p-10},\ b = \frac{e^{\frac{1}{a}}-1}{I-M},\ c = \frac{I-M*e^{\frac{1}{a}}}{I-M} \tag{4.2}$$

A EQ. 4.2 possui três constantes, I, M e p, cujos valores devem ser previamente determinados. I e M determinam as condições de contorno da função. Já a constante p está relacionada à curvatura da função.

A escolha dos valores I e M está relacionada com o enlace utilizado. I deve ter valor igual à capacidade máxima do enlace enquanto que M deve ser igual à MTU. Desta forma, U_{tcp} é máxima quando $Throughput = I$ e mínima quando $Throughput = M$.

A FIG. 4.2 representa a EQ. 4.2 plotada para diferentes valores do parâmetro p. Como pode ser observado, de acordo com a escolha de parâmetro, a curva de variação da satisfação é alterada. Para o valor $p = 17$ a curva aproxima-se da proposta por (RIBEIRO, 2007).

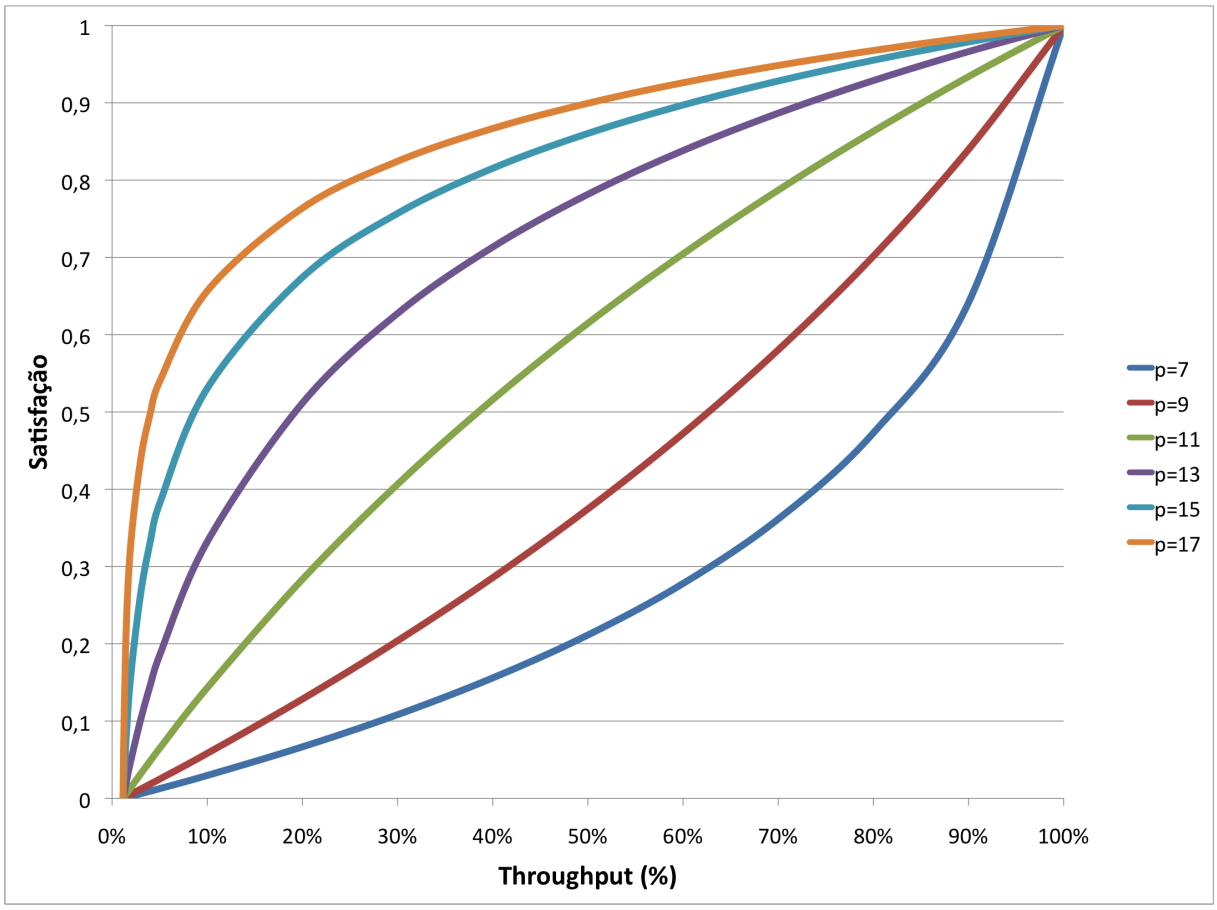

FIG. 4.2: Satisfação X Throughput para diferentes valores de p.

Duas observações importantes cabem à FIG. 4.2. A primeira é referente ao início do eixo *Throughput*. Devido à inclinação da curva, uma pequena variação do *throughput* representa um grande ganho na Satisfação. Quando o *throughput* passa de 0% para 10% de utilização do enlace, a satisfação varia em aproximadamente 65%. Esse dado reflete o ganho de satisfação do usuário que nada transmitia e que passou a transmitir utilizando 10% da capacidade do enlace. Em contrapartida, e esta é a segunda observação, no trecho final do eixo *Throughput*, como a inclinação da curva é pequena, um ganho de 30% de *throughput* representa menos de 10% de aumento de satisfação, pois o usuário já conseguia transmitir informação.

4.2.2 DESUTILIDADE MARGINAL TCP

A desutilidade marginal é um conceito derivado da utilidade marginal que, por sua vez, está associada ao preço que um consumidor está disposto a pagar por unidade adicional

de um bem (VARIAN, 2006).

Matematicamente, a utilidade marginal é obtida calculando-se a derivada da utilidade.

$$U_{marginal} = \frac{dU}{dx} \tag{4.3}$$

Já a desutilidade marginal é o prejuízo que um consumidor percebe por unidade do bem que deixa de consumir.

É importante notar que este prejuízo por unidade de bem não consumido não é linear devido a concavidade da função utilidade proposta tanto por (RIBEIRO, 2007) quanto por (RICHARDS, 1999). Isto significa que o prejuízo quando a utilidade é alta e uma unidade do bem deixa de ser consumida é menor do que quando a utilidade é baixa e a mesma unidade do bem deixa de ser consumida.

4.2.3 TARIFAÇÃO UDP

A tarifação de fluxos UDP deve ocorrer sempre que estes, devido ao seu carácter irresponsável, prejudicarem os fluxos TCP. O objetivo é fazer com que o UDP "pague" pela insatisfação percebida pelo TCP. Porém, deve existir o cuidado de não apenas impor penalidades ao UDP. Algum benefício deve ser oferecido ao UDP para que este não seja banido da rede. Garantir a largura de banda necessária aos diversos fluxos UDP em todos os enlaces utilizados por estes na rede foi a opção escolhida neste estudo.

Através do cálculo da desutilidade marginal TCP, (RIBEIRO, 2007) propõe uma equação para calcular o valor devido.

Para evitar instabilidade no cálculo do preço de transmissão UDP, a desutilidade D do TCP é discretizada em k intervalos. A FIG. 4.3 ilustra a utilidade do TCP para o caso onde $k = 10$.

A arquitetura aqui proposta optou por utilizar $k = 100$ para obter mais faixas de variação de preço, mas sem causar instabilidade.

A função de cálculo do preço para cada enlace assume a seguinte forma:

$$P = D_{k_{tcp}} * C_m * B \tag{4.4}$$

onde Cm é uma constante monetária e B a quantidade de banda reservada. O objetivo da constante monetária é alterar o valor da tarifa de forma linear para ajustá-la ao cenário

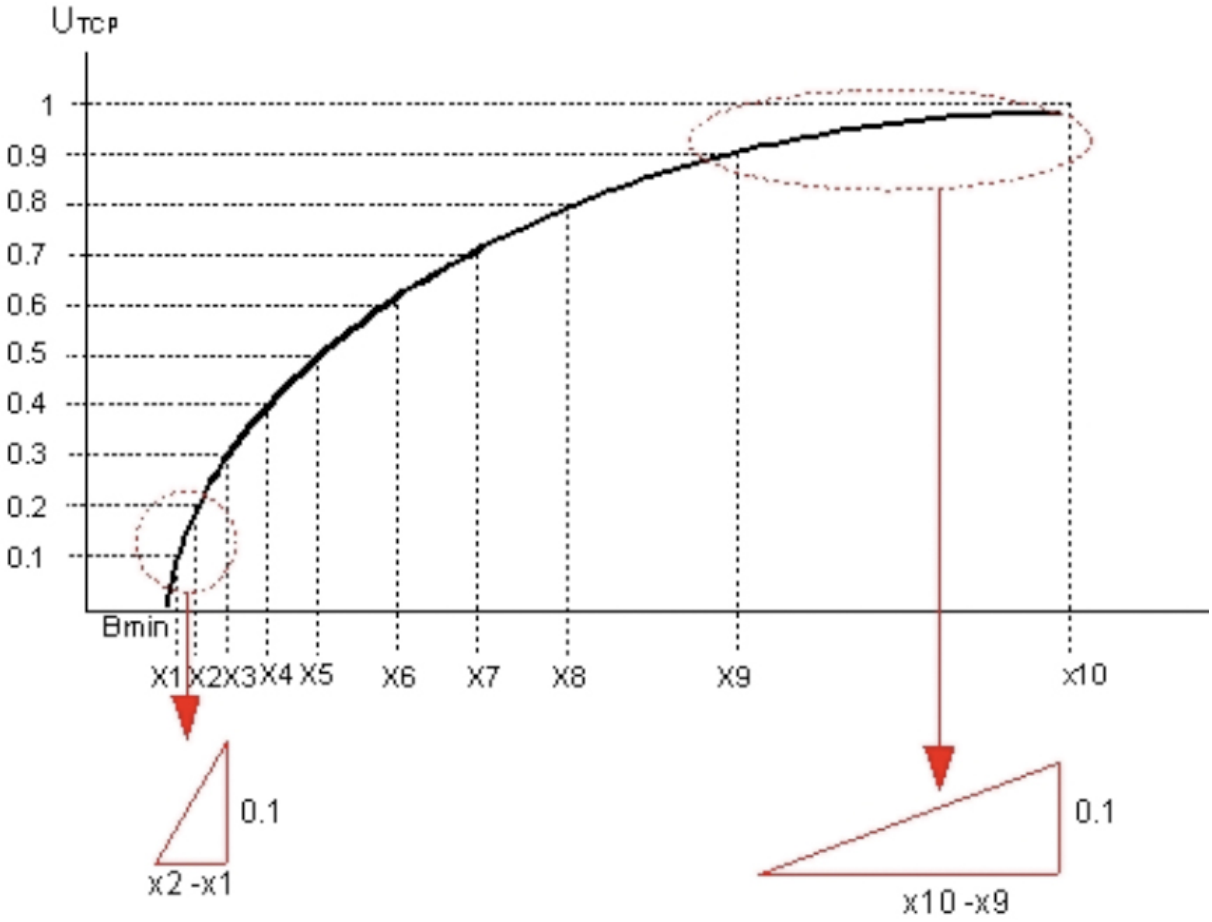

FIG. 4.3: Representação da inclinação da função U_{tcp} na aproximação *Piecewise Linear*. Extraída de (RIBEIRO, 2007)

utilizado. Já quantidade de banda reservada está presente na EQ. 4.4 pois o preço cobrado por uma transmissão deve ser diretamente proporcional à grandeza dos recursos utilizados.

Dentro de cada intervalo a desutilidade marginal é considerada constante e igual a:

$$D_{k_{tcp}} = \frac{\triangle_k U_{tcp}}{\triangle_k x} = 0,01 \triangle_k x^{-1} \tag{4.5}$$

Quanto maior for D_k maior será o preço cobrado do UDP, pois maior é o prejuízo causado ao TCP. Este mecanismo desestimula de forma natural a entrada de fluxos UDP sempre que o tráfego TCP estiver experimentando um baixo desempenho devido ao congestionamento.

Já na situação oposta, valores baixos de D_k ocasionam preços baixos cobrados ao tráfego UDP, estimulando a utilização deste protocolo sempre que não houver congestionamento do tráfego TCP.

4.3 RESERVA E GARANTIA DE RECURSOS

Este subsistema, conforme mostrado na FIG. 4.1, tem a finalidade de prover as garantias necessárias à transmissão dos fluxos UDP através da escolha de um escalonador adequado e de políticas de controle de tráfego.

A arquitetura proposta recomenda a utilização de uma tecnologia que garanta uma baixa quantidade de troca de dados em suas mensagens de configuração dos roteadores da rede administrada, causando pouco impacto no volume de dados que trafegam nos enlaces.

As subseções seguintes descrevem os escalonadores analizados e as políticas de controle de tráfego.

4.3.1 ESCALONADOR

É um algoritmo que escalona os recursos dos enlaces/roteadores entre os fluxos de dados. Existem vários tipos de escalonadores e cabe ao gerente da rede de dados escolher qual melhor atende às políticas estabelecidas para controle da mesma.

Conforme mostrado por (KESHAV, 2001), existem quatro graus de liberdade no desenvolvimento de um escalonador. O primeiro é relativo à quantidade de níveis de prioridade que um escalonador possui. No caso de múltiplas prioridades, as conexões devem ser classificadas de acordo com algum critério e destinadas a um determinado nível de prioridade. Os pacotes de um determinado nível apenas são atendidos quando todas as filas dos níveis superiores estão vazias. Isto permite diminuir o tempo médio de espera na fila do escalonador para conexões de maior prioridade.

O segundo grau de liberdade é relativo ao comportamento de cada nível de prioridade. Cada um pode funcionar de modo conservador ou não. O autor explica que o escalonador "*non-work-conserving*" é aquele que nem sempre atende os pacotes que estão em sua fila esperando serem despachados e só o faz de acordo com um critério específico. Seu propósito é diminuir o *jitter* da rede. Já os escalonadores "*work-conserving*" sempre atendem os pacotes que estão aguardando atendimento. Este último tipo de escalonador apenas fica ocioso quando sua fila está vazia. Seu propósito é oferecer atendimento o mais rápido possível para tráfego do tipo "*best-effort*".

O terceiro corresponde ao grau de agregação em cada nível. Em um extremo, o escalonador utiliza uma variável de estado para descrever todas as conexões, que, desta

maneira, receberão todas a mesma qualidade de serviço. Na outra extremidade, o escalonador mantém uma variável de estado para cada conexão, permitindo o controle de largura de banda e de atraso para cada uma delas.

O quarto e último grau é relativo à ordem de atendimento dentro de cada nível de prioridade. Este grau concebe a idéia de que pacotes previamente marcados devem receber atendimento prioritário dentro de sua fila de prioridade.

A combinação desses quatro graus de liberdade determinam as características e a utilidade de um escalonador.

O arquitetura aqui proposta necessita de um escalonador capaz de reservar parte do enlace para os fluxos UDP para provimento de garantias a estes, uma vez que este protocolo será tarifado por isso, e ao mesmo tempo não permitir a perda de desempenho ou desperdício de recursos. Para escolher o escalonador que melhor atende estas necessidades, as seções seguintes analisam brevemente o comportamento e funcionalidades dos escalonadores mais populares presentes em diversos tipos de roteadores.

4.3.1.1 FIFO - *FIRST-IN-FIRST-OUT*

É o escalonador de mais simples implementação. Nele existe apenas uma fila onde todos os pacotes de todos os fluxos que chegam são colocados em uma mesma fila e despachados exatamente na mesma ordem que chegaram nesta.

A FIG. 4.4 ilustra o funcionamento deste escalonador. Os pacotes dos diversos fluxos estão coloridos com cores diferentes e numerados de acordo com a ordem em que chegam no roteador. Essa mesma ordem é mantida na formação da fila de pacotes que são despachados pela porta do roteador.

Costuma ser o escalonador padrão na maioria dos roteadores devido à sua simplicidade. De acordo com os critérios de (KESHAV, 2001), este escalonador não suporta esquema de priorização de pacotes, é *work-conserving*, possui alto grau de agregação pois trata todas as conexões igualmente e não trabalha com marcação de pacotes.

4.3.1.2 PQ - *PRIORITY QUEUEING*

Este escalonador é utilizado quando fluxos de dados diferentes possuem prioridades diferentes mas compartilham os mesmos recursos, rotas, dentro de uma rede.

Seu algoritmo identifica qual tipo de tráfego tem maior prioridade e despacha este antes daqueles de menor prioridade, mesmo que tenha chegado depois no roteador. Um

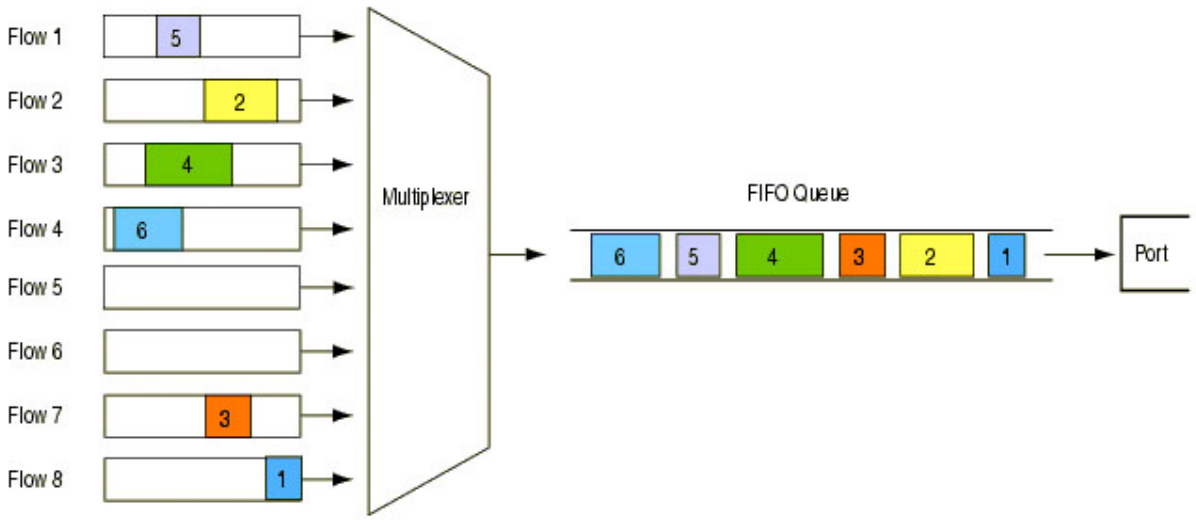

FIG. 4.4: Escalonamento FIFO. Extraída de (BALLIACHE, 2003)

pacote só é servido quando não existir nenhum outro com maior prioridade. Desta forma, a fila de menor prioridade somente é servida quando todas as outras estão vazias, o que pode proporcionar ao tráfego de menor prioridade uma espera demasiadamente longa sempre que tráfegos de maiores prioridades estiverem sendo utilizados intensamente.

A FIG. 4.5 ilustra o funcionamento deste escalonador. Os pacotes que chegam ao roteador são classificados em quatro tipos de prioridade de acordo com um critério previamente estabelecido. Para cada prioridade, uma cor é atribuída aos seus respectivos pacotes para que sejam identificados: verde para prioridade alta, amarelo para prioridade média, azul para prioridade normal e laranja para prioridade baixa. Enquanto existir pacotes verdes no *buffer* de filas, apenas estes serão encaminhados para a fila de transmissão para serem despachados.

De acordo com os critérios de (KESHAV, 2001), este escalonador suporta esquema de priorização de pacotes, é *work-conserving*, possui grau de agregação médio, pois é capaz de classificar as conexões de acordo com características semelhantes que possuírem e oferecer atendimento diferenciado à esses grupos de conexões, mas não trabalha com marcação de pacotes. Uma característica interessante é a possibilidade de customização do classificador de conexões, permitindo variar o grau de agregação utilizado.

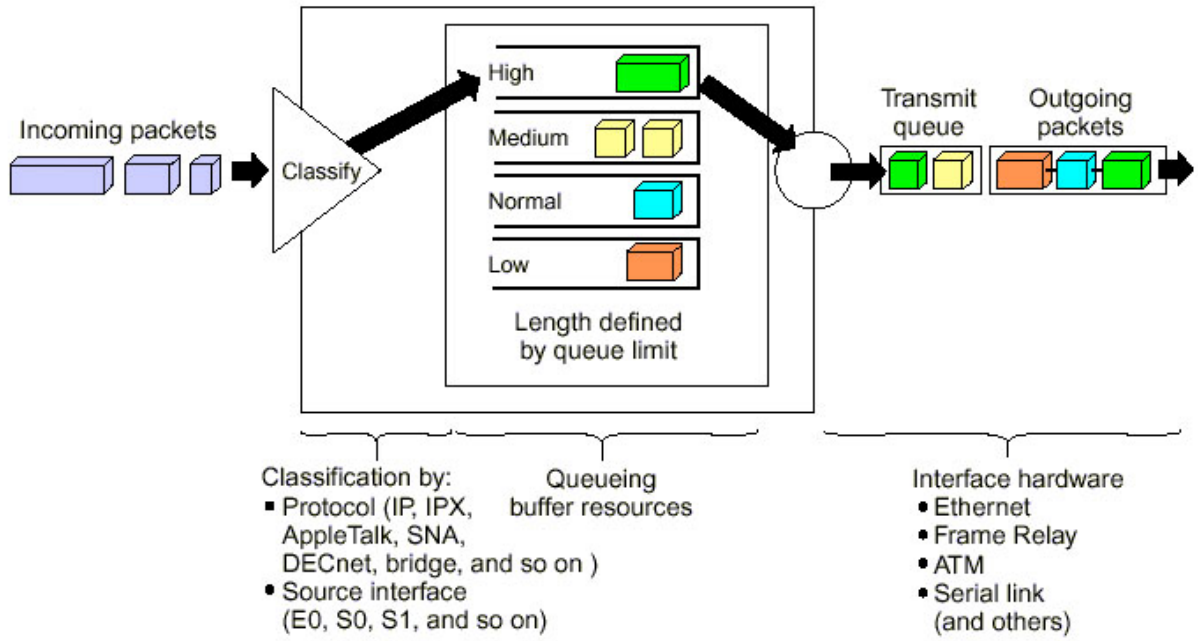

FIG. 4.5: Escalonamento PQ. Extraída de (BALLIACHE, 2003)

4.3.1.3 WFQ - *WEIGHTED FAIR QUEUEING*

Seu algoritmo é um pouco mais complicado que os anteriores já descritos. O escalonador WFQ permite a utilização de diversas filas com atribuição de pesos diferentes a cada uma delas. Seu intuito é dividir a utilização do enlace de forma justa entre os diversos fluxos.

Seu classificador identifica o tipo de tráfego (fluxos) através de atributos contidos no cabeçalho de cada pacote (ip de origem, ip de destino, porta de origem, porta de destino, protocolo, tipo de serviço, etc). Uma função do tipo *hash*, utilizando esses atributos, calcula o peso do pacote e, por conseguinte, a fila a ser designada a este, podendo uma mesma fila receber mais de um tipo de tráfego. Essa característica está evidenciada na FIG. 4.6, que representa o funcionamento deste escalonador. Nesta ilustração, uma mesma fila do *buffer* recebe pacotes de cores diferentes, ou seja, tráfegos diferentes.

O despachante percorre todas as filas do *buffer* de modo *round-robin* transmitindo os pacotes. Se uma fila recebe mais pacotes que outra, isto significa que sua prioridade é menor, pois seus pacotes terão um tempo médio de permanência na fila maior que o da outra.

De acordo com os critérios de (KESHAV, 2001), este escalonador não suporta esquema de priorização de pacotes. Seu objetivo é oferecer igualdade do serviço prestado à todas as conexões. Ele é *work-conserving*, possui grau de agregação médio, porém não

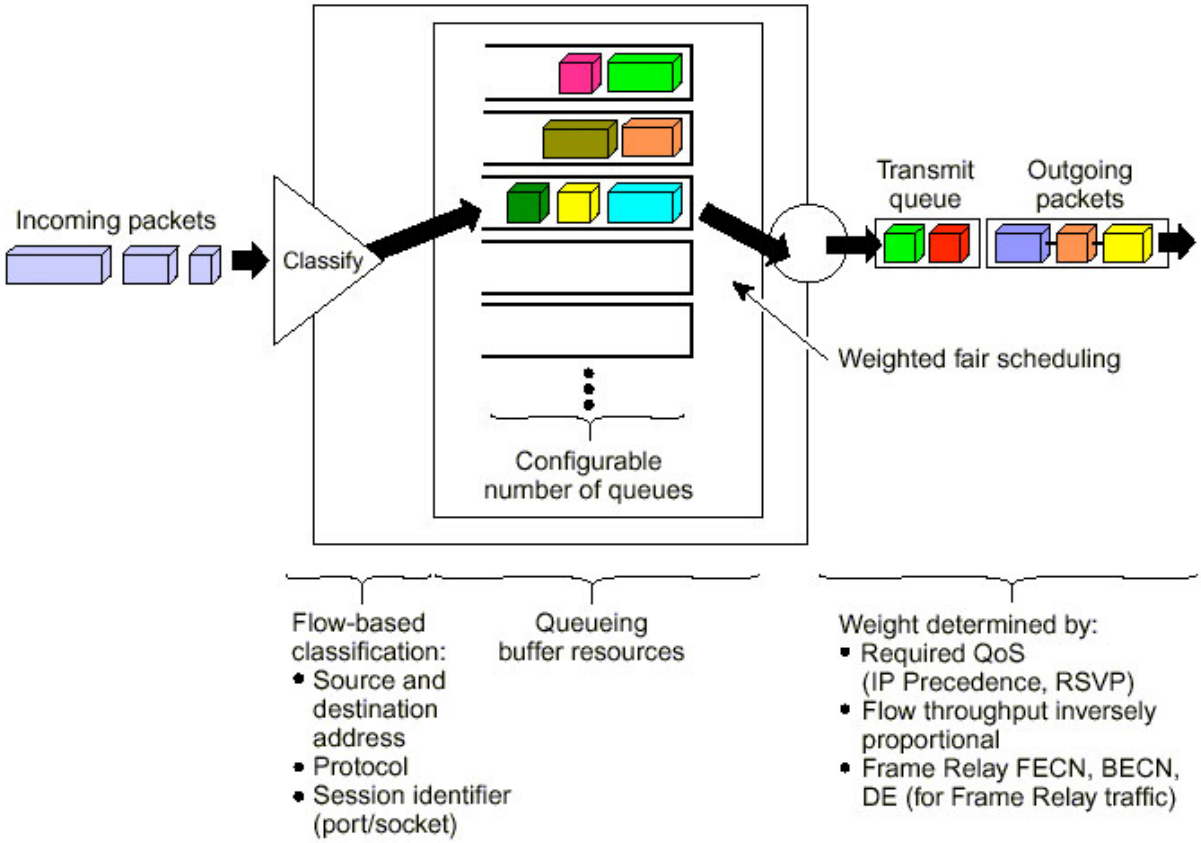

FIG. 4.6: Escalonamento WFQ. Extraída de (BALLIACHE, 2003)

customizável, e não trabalha com marcação de pacotes.

4.3.1.4 CQ - *CUSTOM QUEUEING*

Seu algoritmo permite uma alocação percentual dos recursos disponíveis. Com até 16 filas configuráveis, o classificador separa os pacotes conforme o tipo de tráfego e os designam para as filas correspondentes. O despachante, então, percorre todas as filas de modo *round-robin* transmitindo uma quantidade pré-determinada de pacotes em cada fila. Se não houver pacotes a serem transmitidos em uma determinada fila, o despachante segue imediatamente para a próxima fila.

Uma variação possível e muito útil na configuração deste tipo de escalonador é, ao invés de fixar a quantidade de pacotes a serem transmitidos em cada fila, fixar a quantidade de dados (bytes, por exemplo).

Esse esquema é denominado "*weighted round-robin*" e é através dos pesos atribuídos a cada fila, que pode ser a quantidade de bytes ou de pacotes a serem transmitidos por

rodada, que esse escalonador realiza a priorização do tráfego.

O mecanismo de funcionamento deste escalonador está ilustrado na FIG. 4.7

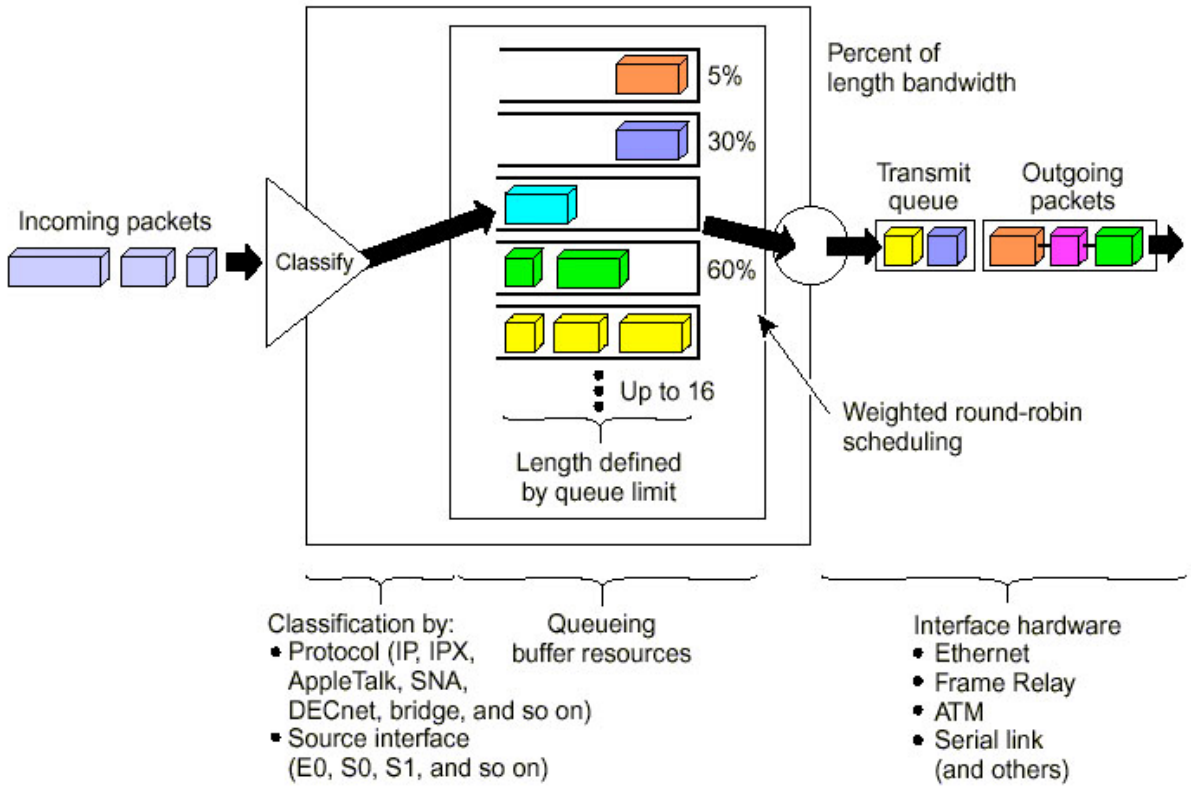

FIG. 4.7: Escalonamento CQ. Extraída de (BALLIACHE, 2003)

De acordo com os critérios de (KESHAV, 2001), este escalonador suporta esquema de priorização de pacotes e é *work-conserving*. Possui grau de agregação médio e customizável, sendo capaz de classificar as conexões de acordo com características semelhantes que possuírem e oferecer atendimento diferenciado à esses grupos de conexões. E não trabalha com marcação de pacotes.

4.3.1.5 ESCALONADOR ESCOLHIDO

O escalonador escolhido para figurar no subsistema proposto foi o CQ (*custom queueing*). Apesar de possuir graus de liberdade semelhantes ao PQ (*priority queueing*), o escalonador CQ permite a divisão percentual de utilização de cada enlace conforme o tipo de tráfego através de seu esquema *weighted round-robin*, característica necessária para o funcionamento da arquitetura. O experimento a seguir comprova sua eficácia.

A topologia adotada foi a mesma da FIG. 3.1 e a obtenção e a análise dos dados foi

realizada, novamente, com o auxílio da ferramenta ManageEngine NetFlow Analizer (ADVENTNET). Foram gerados um fluxo UDP e três fluxos TCP durante todo o experimento. A taxa de envio de dados de cada um dos quatro fluxos foi escolhida de modo a congestionar, todas juntas, o enlace de 1 Mbit/s. O fluxo UDP gerado foi aproximadamente 90% da capacidade do enlace enquanto que os três fluxos TCP tiveram sua taxas de envio variando em função da disponibilidade do enlace, mas sempre procurando atingir o limite de 100% da capacidade deste. No decorrer do experimento, o percentual de reserva do escalonador CQ para tráfego UDP foi alterado oito vezes. As quatro primeiras diminuíram a reserva de 50% para 10%. As quatro alterações seguintes elevaram o parâmetro de reserva de tráfego UDP, reestabelecendo-o em 50%.

A FIG. 4.8 representa o experimento descrito. A linha azul representa o tráfego UDP gerado, sempre em torno de 90% da capacidade do enlace. Apesar de gerado, parte desse tráfego não atravessa o enlace devido ao parâmetro de reserva do escalonador CQ. A linha vermelha representa *goodput* do tráfego UDP, ou seja, os pacotes que efetivamente atravessaram o enlace e foram recebidos pela estação de trabalho receptora. A linha verde representa o *goodput* agregado dos três fluxos TCP. Já linha roxa representa o *goodput* agregado de tráfego TCP e UDP.

A análise da FIG. 4.8 permite duas observações. A primeira é relativa a eficácia do escalonador CQ. Apesar do tráfego UDP gerado ser sempre aproximadamente 90%, apenas trafega no enlace a quantidade reservada pelo escalonador. A segunda observação é relativa ao comportamento dos fluxos TCP. O *goodput* agregado dos três fluxos TCP é sempre complementar ao *goodput* do fluxo UDP, totalizando junto deste 100% de utilização do enlace. Fica evidenciado o caráter adaptativo do protocolo TCP ao evitar congestionamento sem permitir que o enlace seja subutilizado.

4.3.2 CONTROLE DE TRÁFEGO

A reserva de recurso para fluxos UDP é feita com a utilização do escalonador CQ. Já a moderação destes fluxos deve ser realizada através de regras de bloqueio nos respectivos *firewalls* com a finalidade de não permitir tráfego UDP não autorizado (não pagante).

A arquitetura funciona bloqueando todo o tráfego UDP na rede e criando regras específicas, dinamicamente, para permitir a transmissão dos fluxos UDP solicitados pelos clientes e aceitos pelo Bandwidth Broker.

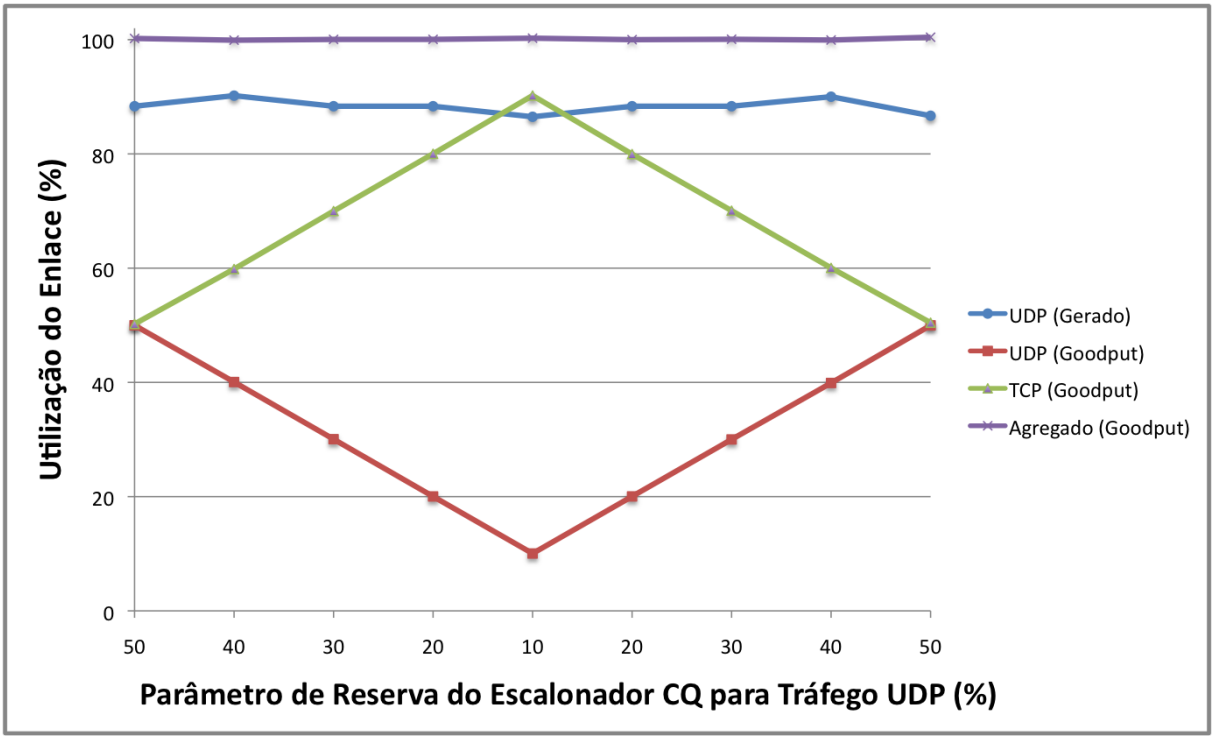

FIG. 4.8: Impacto no Tráfego de um Enlace Devido à Variação dos Parâmetros do Escalonador Custom Queueing.

4.4 ESTADO DOS ENLACES

Este subsistema (ver FIG. 4.1) é responsável por conhecer o estado de cada enlace dentro da rede, o que é feito utilizando a tecnologia de fluxos incorporada nos roteadores. Os protocolos de fluxo exportam informações detalhadas sobre os fluxos de dados que trafegam em determinado ativo da rede.

Uma grande vantagem para a arquitetura em utilizar este esquema é a escalabilidade alcançada. O Bandwidth Broker não precisa solicitar constantemente informações aos roteadores para conhecer o estado dos enlaces. Ele apenas recebe os *exports* dos roteadores com as informações sobre os fluxos e as analisa.

Inicialmente desenvolvida pela Cisco Systems, a tecnologia recebeu o nome de NetFlow e, apesar de proprietária, seu código é aberto. Este fato contribui para que hoje diversas empresas possuam soluções semelhantes implementadas tanto em *hardware* quanto em *software*.

A TAB. 4.1 relaciona algumas das tecnologias de fluxos existentes com os fabricantes de roteadores que as adotaram em seus equipamentos.

Tipo de Fluxo	Fabricante
NetFlow	Cisco, 3Com, Adtran, Riverbed, Enterasys, Extreme Networks, MikroTik, Juniper Networks, Foundry Networks
sFlow	AlaxalA Networks, Alcatel, Allied Telesis, Comtec, Force10 Networks, Foundry Networks, Hewlett-Packard, Hitachi, NEC
J-Flow	Juniper Networks
NetStream	Huawei, H3C
IPFIX	Nortel
rFlow	DD-WRT

TAB. 4.1: Tipos de fluxos e seus fabricantes. Fonte: (ADVENTNET)

A arquitetura aqui proposta recomenda a utilização de IPFIX como tecnologia de informação de fluxos, já que ela é uma proposta do IETF (*Internet Engineering Task Force*) para a padronização da tecnologia. Por esse motivo, é esperado que no futuro a maioria dos equipamentos de rede tenham suporte à esse protocolo. A proposta do protocolo IPFIX é baseada na versão 9 do Cisco NetFlow.

Como a padronização da tecnologia de informação de fluxos ainda não é uma realidade, esta arquitetura optou por utilizar o NetFlow por ser a tecnologia mais antiga, consistente e difundida.

4.4.1 NETFLOW

É a tecnologia de informação de fluxos desenvolvida pela Cisco Systems. Cada ativo de rede que tiver essa tecnologia habilitada exporta, para um servidor específico através de datagramas UDP, as informações sobre os fluxos que trafegam pelas interfaces do equipamento. Dependendo da intensidade de utilização do roteador, o tamanho do pacote contendo as informações dos fluxos pode variar bastante pois o tamanho depende da quantidade de fluxos que foram encapsulados no mesmo datagrama. A FIG. 4.9 contém um exemplo de datagrama de um *export* NetFlow e a FIG. 4.10 mostra as informações a respeito de um determinado fluxo contidas no registro de fluxo encapsulado dentro de um *export*.

A habilitação da tecnologia em um equipamento é feita por interface. Isto significa que um roteador não precisa ter todas as suas interfaces utilizando a tecnologia de informação de fluxos e, conseqüentemente, gerando tráfego através dos *exports* para o servidor coletor das informações.

O NetFlow possui 5 versões, sendo a 5 e a 9 as mais típicas. A versão 1 é a mais antiga

e menos popular, sendo raramente utilizada. Já a versão 5 adcionou informações sobre os sistemas autônomos em que os fluxos trafegam e o número sequencial do fluxo gerado. A versão 8 é utilizada quando o roteador possui a funcionalidade de agregação de fluxos ativada.

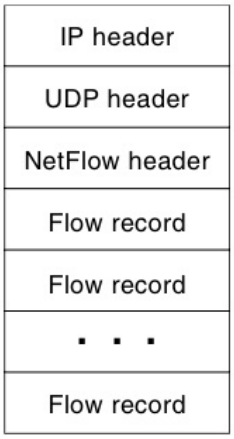

FIG. 4.9: Datagrama de um *export* NetFlow nas versões 1, 5, 7 ou 8. Extraída de (CISCO, 2006)

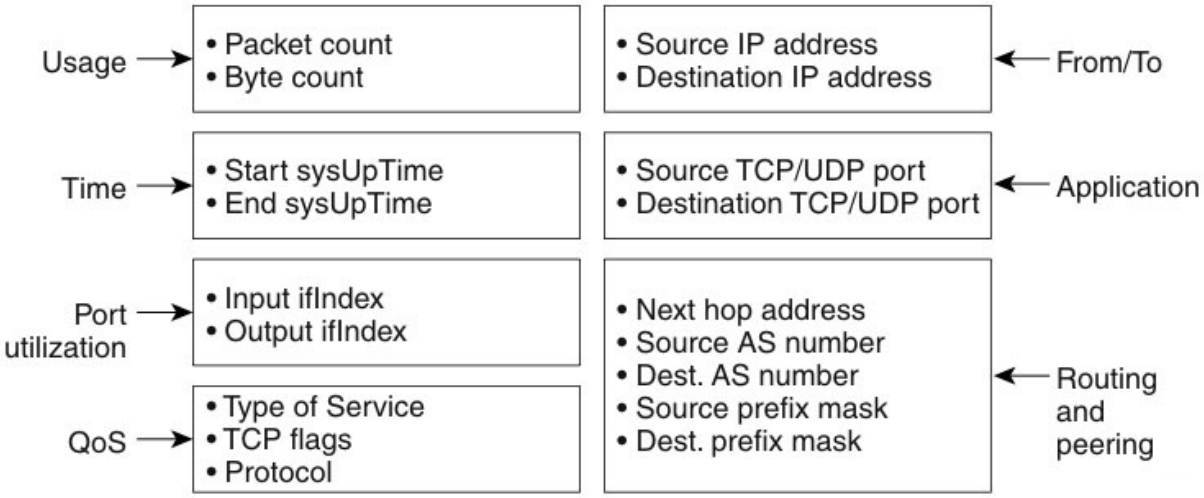

FIG. 4.10: Formato de um registro NetFlow na versão 5. Extraída de (CISCO, 2006)

A versão 9 possui o grande diferencial de ser baseada em *templates*, permitindo a customização dos formatos dos campos de informações contidos nos fluxos exportados (CISCO, 2006). Nesta versão, qualquer informação da camada 2 (enlace) até a camada 7 (aplicação) pode ser transmitida, além de possuir suporte a IPv6 e multicast.

4.5 INFORMAÇÃO

A função do subsistema informação (ver FIG. 4.1) é armazenar as informações sobre o estado de cada enlace da rede para posterior análise, bem como informações sobre os roteadores, usuários e cobranças.

Este subsistema se materializa através da utilização de um banco de dados relacional que deve conter três tabelas. A primeira é utilizada para armazenar todas as informações sobre os fluxos de dados que trafegam em todos os enlaces da rede administrada: nome (ip) do roteador, data do recebimento do fluxo no Bandwidth Broker, versão do protocolo de fluxo utilizado, ip de origem, ip de destino, porta de origem, porta de destino, interface de origem, interface de destino, próximo *hop*, protocolo, quantidade de *bytes* transmitidos, quantidades de pacotes transmitidos, data de início do fluxo, data de término do fluxo, e demais outras informações dependendo da versão do protocolo de fluxos utilizado na implementação da arquitetura.

Através de consultas específicas, que serão vistas no capítulo 5, é possível conhecer a utilização de cada enlace feita pelo tráfego TCP e reconhecer se e o quão congestionado eles estão.

A segunda tabela deve armazenar as informações da topologia da rede. Cada linha da tabela representa uma interface ativa de um roteador, ou seja, uma extremidade de um enlace. Para cada interface é armazenado o seu nome, o seu ip, o seu número (*ifIndex*), a sua velocidade, o seu percentual de reserva para tráfego UDP, o nome do roteador em que ela está presente e o nome do roteador da outra extremidade do enlace.

A terceira tabela é a mais simples e deve conter os dados para posterior cobrança pela utilização da reserva do enlace para trafegar UDP. Cada linha contém o nome do usuário, o valor devido e se o mesmo já foi pago. Ao receber a tarifa devida, o sistema deve atualizar a respectiva coluna da entrada correspondente informando que a dívida foi paga.

5 IMPLEMENTAÇÃO DA ARQUITETURA PROPOSTA

Para implementação da arquitetura proposta foi necessário mapear as funcionalidades de todos os subsistemas em procedimentos específicos, que por sua vez podem ser reorganizados e agrupados em macroprocessos.

A figura 5.1, que será detalhada nas seções seguintes, permite visualizar os atores e os macroprocessos do sistema.

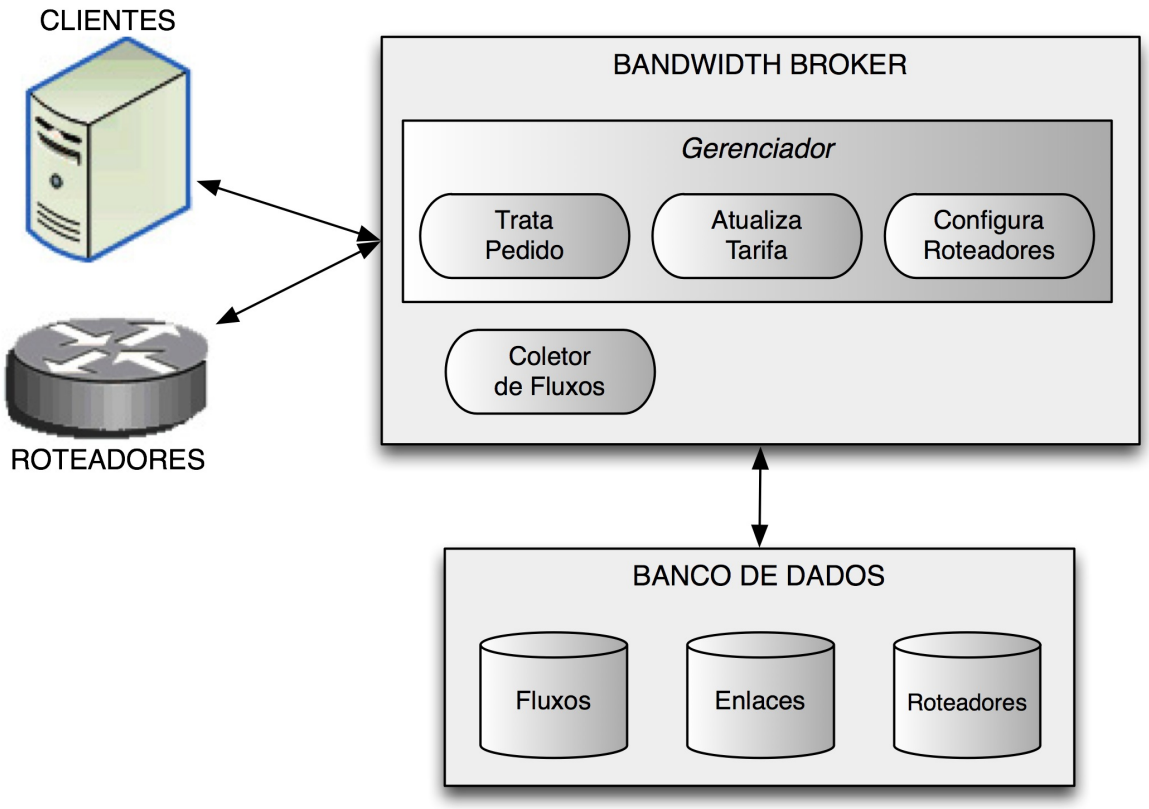

FIG. 5.1: Visão do sistema proposto.

5.1 LINGUAGEM UTILIZADA

Devido a facilidade e domínio por parte do autor, a linguagem C foi escolhida para implementação computacional da arquitetura proposta.

5.2 REUSO DE SOFTWARE LIVRE

Com o intuito de evitar desperdício de tempo, optou-se por aproveitar software com livre licença de uso na implementação das funcionalidades requeridas pela arquitetura proposta, sempre que possível e conveniente para o presente trabalho.

5.3 BANDWIDTH BROKER

O Bandwidth Broker compreende dois *softwares* com funções específicas. O primeiro é o Coletor de Fluxos, responsável por receber e armazenar os *exports* com as informações sobre o tráfego nos enlaces em um banco de dados para posterior análise. O segundo é o Gerenciador, responsável por analisar o estado da rede através de consultas às informações salvas no banco de dados, verificar, processar e aceitar pedidos de transmissão UDP e reconfigurar os roteadores.

5.3.1 COLETOR DE FLUXOS

Foi realizada uma pesquisa sobre coletores de fluxo com código livre que possam ser aproveitados como parte da implementação do Bandwidth Broker. Dentre os diversos coletores estudados, os *softwares* NEye (PERSICO, 2005), Flowd (MILLER, 2008) e F.L.A.V.I.O. (VILLANUSTRE, 2002) foram os que apresentaram características mais próximas das desejadas para serem aproveitados.

O escolhido foi o NEye por apresentar maior organização do código fonte, utilização da mesma versão do NetFlow dos roteadores do Laboratório de Redes da Seção de Engenharia de Computação do IME (NetFlow versão 5), utilização de *threads* para captura dos fluxos e compatibilidade de armazenamento dos dados no banco de dados MySQL 5.0 (SUNMICROSYSTEMS, 2007), parte integrante do sistema operacional openSuse 11.0 instalado nas estações de trabalho do referido laboratório.

A única funcionalidade desejada e não implementada pela última versão do NEye (v. 1.0.1), disponível até a presente data, é o armazenamento das interfaces dos roteadores correspondentes ao fluxos. Tal funcionalidade foi adcionada com algumas alterações no código fonte do coletor de fluxos NEye.

Este *software* teve sua versão 1.0.1 alterada para permitir o armazenamento das interfaces de entrada e saída dos roteadores correspondentes a determinado fluxo.

O banco de dados utilizado pelo NEye também foi alterado para permitir o armazenamento de dados referentes à rede administrada. Foi criada uma tabela adicional para armazenar as informações e assim permitir quantizar as conexões ativas e *throughput* TCP por enlace através da análise dos fluxos recebidos.

5.3.2 GERENCIADOR

O *software* gerenciador, de modo geral, realiza consultas ao banco de dados para conhecer o estado atual dos enlaces da rede administrada e, com os dados obtidos, poder calcular as tarifas de reserva de recursos para tráfego UDP e, se for o caso, reconfigurar os roteadores para acomodar os fluxos deste protocolo.

O fluxograma da FIG. 5.2 representa o funcionamento do Gerenciador.

O gerenciador, através do módulo "*Aceita Conexão de Novo Pedido*", permanece "escutando", em uma porta previamente combinada, a novos pedidos de clientes que desejam transmitir tráfego UDP. Os pedidos são feitos através de conexões TCP. Sempre que recebe um novo pedido, o gerenciador passa para o módulo *"Fork"* e dividi-se em dois processos. O processo pai volta a "escutar" novos pedidos (conexões TCP) enquanto o processo filho passa para o módulo "*Recebe Pedido*" para então tratá-lo.

O usuário informa em seu pedido o IP origem e o IP destino de sua transmissão, o tempo de duração desta, a largura de banda desejada e o custo máximo que está disposto a pagar pela transmissão. É importante salientar que mesmo que o cliente ofereça um valor maior que a tarifa vigente para a transmissão UDP solicitada, é cobrada a tarifa vigente, que é menor.

O gerenciador então, através do módulo "*Atualiza Tarifa*", calcula a rota e o custo da transmissão. Como a arquitetura não prevê alteração de rotas nos ativos da rede, este módulo determina as rotas da mesma forma que os roteadores da rede. Todos eles estão configurados para utilizar o protocolo OSPF onde todas as rotas possuem pesos de acordo com a determinação Cisco, $peso = \frac{10^8}{Enlace}$. Após determinar a rota através do algoritmo *Dijkstra* e verificar a disponibilidade de banda dos enlaces para acomodar a nova reserva, o módulo atualiza as respectivas tarifas dos enlaces, somando-as em seguida para encontrar o custo da transmissão. Uma maneira de tornar a implementação mais flexível, mas não adotada por não ser necessária para homologar a arquitetura proposta, é importar a tabela de *forwarding* de cada roteador para então calcular a rota ao invés de utilizar o algoritmo de *Dijkstra*.

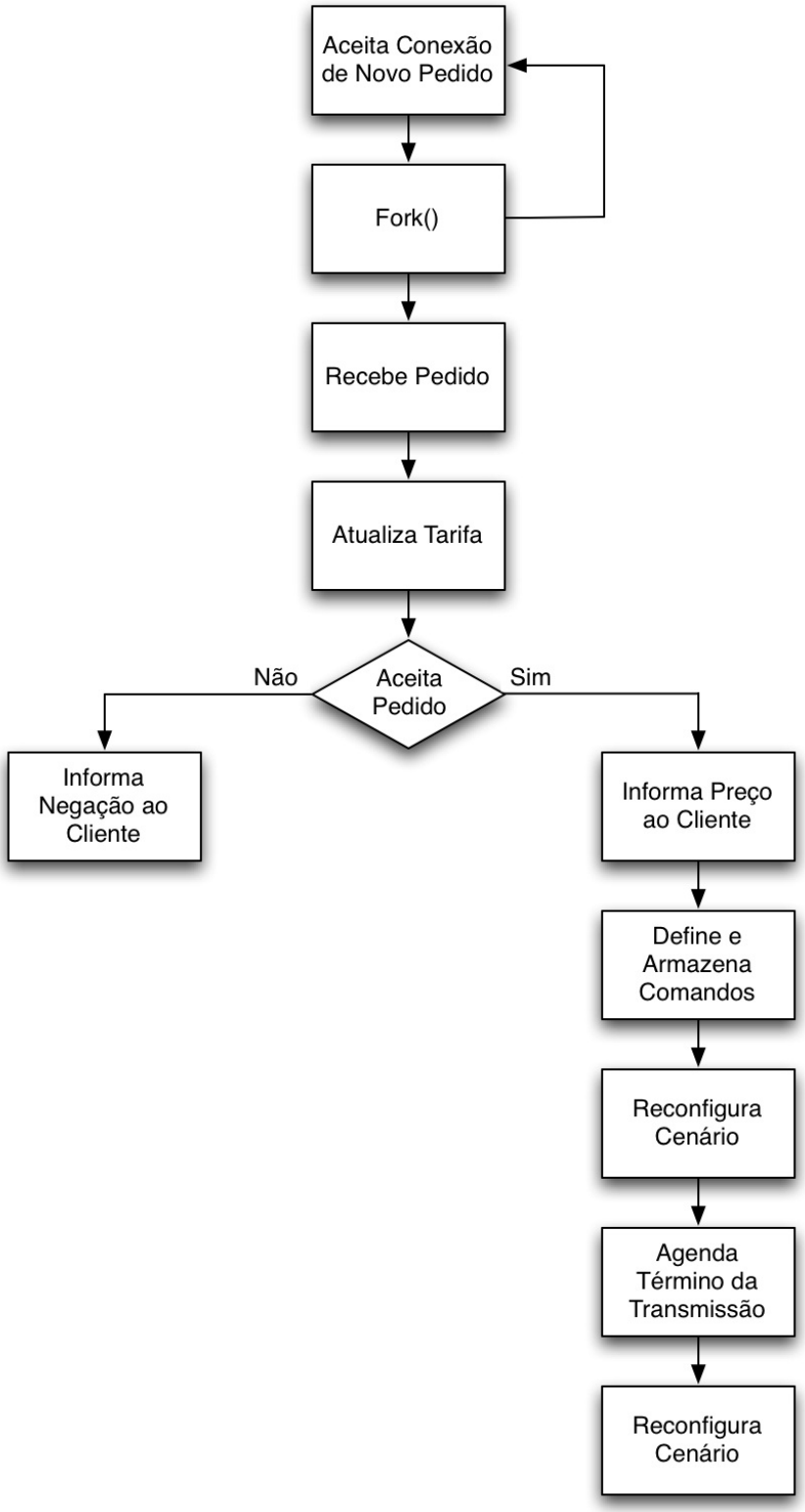

FIG. 5.2: Fluxograma do módulo Gerenciador do Bandwidth Broker.

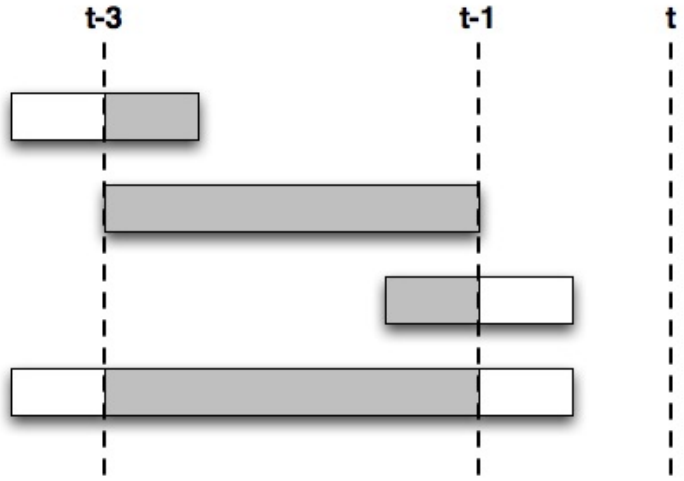

FIG. 5.3: Intervalo de tempo em minutos utilizado para obter informações sobre o estado de um determinado enlace.

A disponibilidade dos enlaces é verificada obedecendo ao critério de não ultrapassar o limite máximo de reserva UDP do sistema. Se pelo menos um enlace da rota calculada não tiver recursos disponíveis para acomodar o pedido, este é negado. O limite máximo de reserva UDP é ajustável pelo administrador da rede, sendo pré-configurado em 90%. Desta forma, no mínimo 10% de cada enlace é garantido para as transmissões TCP a qualquer tempo.

Já o cálculo da tarifa de cada enlace é realizado considerando o histórico de seus respectivos fluxos dentro da janela temporal de 1 à 3 minutos decorridos mais recentemente. Este 1 minuto mais recente é desconsiderado para evitar informações incompletas a respeito do estado do enlace pois a tecnologia de fluxos adotada pode levar até 1 minuto para consolidar as respectivas informações e exportá-las para o Coletor de Fluxos.

A FIG. 5.3 ilustra as quatro situações possíveis em que um determinado fluxo pode estar localizado no tempo em relação ao instante t da atualização das informações de estado dos enlaces. Para cada situação, uma consulta ao banco de dados deve ser realizada. O instante t deve ser o mesmo para todas as quatro consultas para evitar a obtenção de informações erradas quando o volume de chegada de fluxos ao Coletor for intenso. Apenas a parte mais escura de cada fluxo da figura é considerada no cálculo de tráfego do respectivo enlace.

Ainda com relação à tarifa, o sistema possui uma variável denomidada Constante Monetária (C_m) capaz de variar linearmente a tarifa cobrada pelo sistema. Esta variável

é pré-configura com valor unitário e pode ser alterada pelo administrador da rede.

Se a disponibilidade financeira do usuário for suficiente, o Gerenciador aceita o pedido e informa a tarifa cobrada através do módulo "*Informa Preço ao Cliente*". Caso contrário, o Gerenciador informa a negação do serviço e a tarifa atual através do módulo "*Informa Negação ao Cliente*" e depois encerra o processo filho.

No caso em que o pedido é aceito, o processo filho define quais comandos devem ser aplicados nos roteadores para que a rede seja reconfigurada para acomodar o novo fluxo UDP. Esta tarefa é realizada pelo módulo "*Define e Armazena Comandos*". Esses comandos incluem a alteração do parâmetro de configuração da fila UDP do escalonador CQ, para reservar recursos, e regras para permitir o tráfego (ver Apêndice 9.1.3). Cabe uma ressalva neste ponto. Nenhum tipo de controle de execução dos processos paralelos foi implementado e, por isso, é possível que um processo obtenha um valor desatualizado do percentual já reservado de um enlace. Este caso ocorre quando um processo lê a variável de reserva antes que outro processo que já tenha lido a atualize. Porém, considerando que o emprego da arquitetura sugerida neste trabalho destina-se à ambientes controlados, onde a taxa de pedidos de transmissão UDP é conhecida, e que a implementação desta arquitetura possui objetivo acadêmico, a ocorrência deste caso foi desconsiderada.

Como o objetivo da dissertação é propor e verificar a arquitetura, a implementação do controle de reserva de recursos também foi simplificada. Ao reservar um percentual dos recursos de um enlace, os demais enlaces dos dois roteadores conectados pelo primeiro também sofrem reservas de recursos. Isto acontece porque os mesmo parâmetros de configuração do escalonador CQ são aplicados em todas as interfaces dos roteadores.

Com relação ao processo de reconfiguração dos roteadores, pensou-se utilizar SNMP uma vez que essa tecnologia gera pouco *overhead* na rede e é bastante popular. Porém, os equipamentos do Laboratório de Redes utilizados nos experimentos são todos Cisco, não possuidores das MIBs necessárias. Optou-se, então, por aplicar os comandos de reconfiguração dos roteadores através de *scripts* executados paralelamente e aplicados pelo módulo "*Reconfigura Cenário*". O paralelismo na execução melhora o desempenho diminuindo o tempo de configuração dos roteadores.

O processo filho espera o término do tempo reservado pelo cliente através do módulo "*Agenda Término da Transmissão*" e então reconfigura a rede através do módulo "*Reconfigura Cenário*", desfazendo as alterações de reserva de recursos e permissão do tráfego específico e encerrando o processo filho.

5.4 CLIENTE

O *software* Cliente é o meio de comunicação utilizado para transmitir ao Bandwidth Broker, mais especificamente ao Gerenciador, as informações da transmissão UDP desejada pelo usuário.

Como já explicado na seção 5.3.2, o Gerenciador abre uma conexão TCP para receber os dados do cliente que são enviados pelo *software* Cliente.

O *software* transmite as seguintes informações:

1º- IP de origem

2º- IP de destino

3º- Duração da transmissão em segundos

4º- Langura de banda desejada em bits/s

5º- Preço máximo aceitável em unidade monetária

Após transmitidas, o *software* permanece aguardando duas respostas do Gerenciador:

1ª- Resposta (sim ou não)

2ª- Tarifa

A primeira é se a reserva de recursos para a transmissão UDP pretendida foi aceita ou não. A segunda é a tarifa em vigor no momento. Esta segunda resposta tem sua interpretação dependente da primeira resposta. No caso em que a transmissão foi aceita, ela significa o valor que o cliente terá que pagar pelo seu pedido. No caso em que a transmissão não foi aceita, ela siginifica a tarifa mínima necessária para realizar a transmissão.

6 TESTES E RESULTADOS

O objetivo dos testes realizados é comprovar que a arquitetura proposta:

a) evita o congestionamento do tráfego TCP;

b) gera garantias ao tráfego UDP agregado;

c) desestimula a entrada de novos fluxos UDP quando o tráfego TCP está experimentando baixo desempenho;

d) permite ao administrador da rede fixar um percentual de utilização do enlace para tráfego UDP através do controle de tarifa;

e) funciona em um cenário com mais de um enlace.

Para tanto, foram realizados três testes com emprego de metodologia semelhante à dos testes executados nos capítulos anteriores. A obtenção dos dados de utilização dos enlaces foi feita através de análise do banco de dados do *Bandwidth Broker*. Através de quatro *selects* obedecendo os limites mostrados na FIG. 5.3, foi possível obter o *goodput* dos tráfegos UDP e TCP. Estas *queries* são idênticas às utilizadas no módulo "*Atualiza Tarifa*" do *software* Gerenciador (ver Apêndice 9.1.1) para o caso do tráfego TCP. Para o caso UDP, basta alterar o valor do código de identificação de protocolo da tecnologia de fluxos. Ao invés de 6 (protocolo TCP), deve-se utilizar 17 (protocolo UDP).

Nos dois primeiros testes foi utilizado o mesmo cenário da FIG. 3.1, onde um "corte" do esquema de rede do Laboratório de Redes da Seção de Engenharia de Computação do IME evidencia as duas subredes utilizadas nos testes e o enlace de 1 Mbits/s analizado.

No primeiro teste, cujos resultados estão contidos na FIG. 6.1, pedidos de transmissão UDP são feitos a cada 5 minutos apenas durante os primeiros 90 minutos e depois não mais. Cada pedido solicita a alocação de 50 kbits/s por um período de 90 minutos. A linha vermelha representa o *goodput* percentual em relacão à capacidade do enlace obtido pelo tráfego TCP agregado. A linha azul representa o *goodput* total do enlace, ou seja, a soma do *goodput* do tráfego UDP agregado e do tráfego TCP agregado. Observando a linha azul é fácil perceber que a capacidade do enlace foi bem aproveitada. A média de utilização foi de 99% e o menor índice foi de 96%.

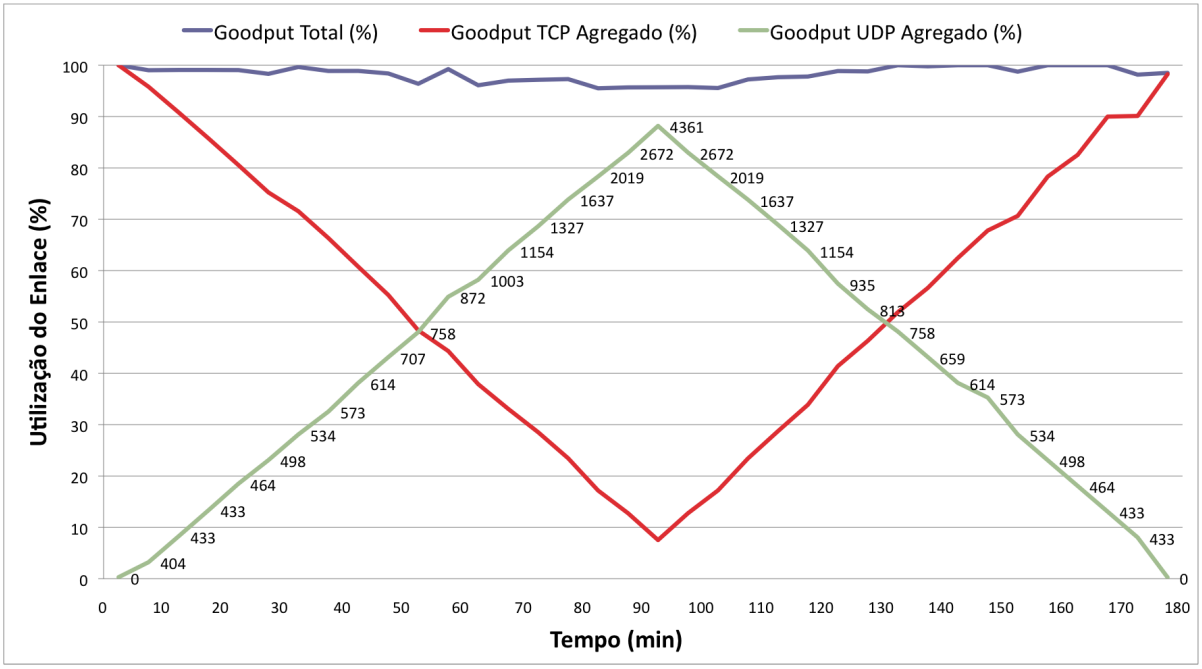

FIG. 6.1: Variação da Tarifa conforme a Utilização do Enlace.

Já a linha verde da FIG. 6.1 possui duplo significado. Além de indicar o *goodput* percentual do tráfego UDP em relação à capacidade do enlace, ela também indica a tarifa vigente para acomodar no enlace uma nova transmissão padrão de 50 kbits/s durante 90 minutos utilizando o protocolo UDP. Quanto menor o rendimento do tráfego TCP causado pela maior utilização de tráfego UDP, maior é a tarifa vigente.

Como durante os primeiros 90 minutos do teste novos pedidos de transmissão UDP são feitos, a tarifa e o *goodput* do tráfego UDP sempre aumentam no decorrer do tempo dentro deste período. Após os 90 minutos iniciais as transmissões passam a terminar, o que diminui o *goodput* do tráfego UDP e, consequentemente, aumenta o *goodput* do tráfego TCP e diminui a tarifa.

Quando o tráfego UDP ocupa 55% do enlace a tarifa (872 u.m.) é mais do que o dobro do que quando a ocupação é de 15% (433 u.m.). No caso em que o tráfego UDP ocupa 90% do enlace a tarifa (4.361 u.m.) é 10 vezes maior do que quando a ocupação é de 15%. Pensando em valores absolutos, a diferença entre as tarifas neste último caso é de 3.928 u.m. Caso o tráfego atravessasse cinco enlaces, todos iguais e na mesma situação, ao invés de apenas um, a diferença entre as tarifas seria de 19.640 u.m.

Fica evidenciado que os três primeiros objetivos foram alcançados pois: o tráfego TCP, durante todo o teste, sempre teve pelo menos 10% do enlace disponível; garantia

de largura de banda foi dada ao tráfego UDP agregado reduzindo o tráfego TCP; e, nos casos em que o tráfego TCP está experimentando baixo desempenho, a tarifa torna-se muito elevada.

No segundo teste, pedidos de transmissão UDP são feitos a cada 5 minutos durante todo o tempo de duração. Cada pedido de transmissão UDP solicita a alocação de 50 kbits/s durante 50 minutos ao invés de 90 minutos como no primeiro teste. Uma outra alteração em relação ao teste anterior é que foi estipulado que nenhum usuário seria capaz de pagar mais do que 330 u.m. pela alocação de 50 kbits/s durante 50 minutos.

Na FIG. 6.2, que contém os resultados obtidos no segundo teste, a linha azul representa o *goodput* total do enlace. Novamente observa-se a quase máxima utilização do enlace durante o teste. A linha vermelha indica o *goodput* percentual do tráfego TCP agregado em relação à utilização do enlace. A linha verde possui o mesmo significado, porém representa o tráfego UDP agregado.

Para cada situação de utilização do enlace, o eixo horizontal indica o valor da tarifa vigente no sistema para acomodar uma nova transmissão padrão de 50 kbits/s durante 50 minutos utilizando o protocolo UDP.

O *goodput* UDP aumenta durante o início do teste e depois permanace constante. O aumento é devido a entrada de novas transmissões UDP. Com a tarifa máxima aceitável por cada usuário estipulada em 330 u.m., o sistema consegue acomodar 8 conexões. Para acomodar a nona conexão o usuário teria que pagar 341 u.m. pela transmissão, extrapolando o valor aceitável. Dessa forma, o trecho seguinte da curva de *goodput* UDP permanece constante. Sempre que uma transmissão UDP termina, a tarifa diminui e uma nova é aceita.

Através da FIG. 6.2, observa-se que a hipótese de uma tarifa máxima aceitável de 330 u.m. resultou em um tráfego UDP menor do que 35% da capacidade do enlace. Fazendo o raciocínio inverso, um gerente de rede é capaz de impor um percentual máximo de utilização de tráfego UDP através da regulagem da variável Constante Monetária (C_m) do Bandwidth Broker e de uma pesquisa de opinião entre os usuários da rede para conhecer a tarifa máxima aceitável. Desta forma, o quarto objetivo da arquitetura é alcançado.

Para o terceiro teste, o cenário utilizado está ilustrado na FIG. 6.3. Neste teste são utilizados dois enlaces seriais de 1 Mbit/s com MTU de 1500 bytes. As transmissões são originadas pelo host 192.168.1.1, pertecente à subrede 192.168.1.0, e destinadas ao host 192.168.2.1, pertecente à subrede 192.168.2.0. Tanto a rota de ida quanto a de volta são

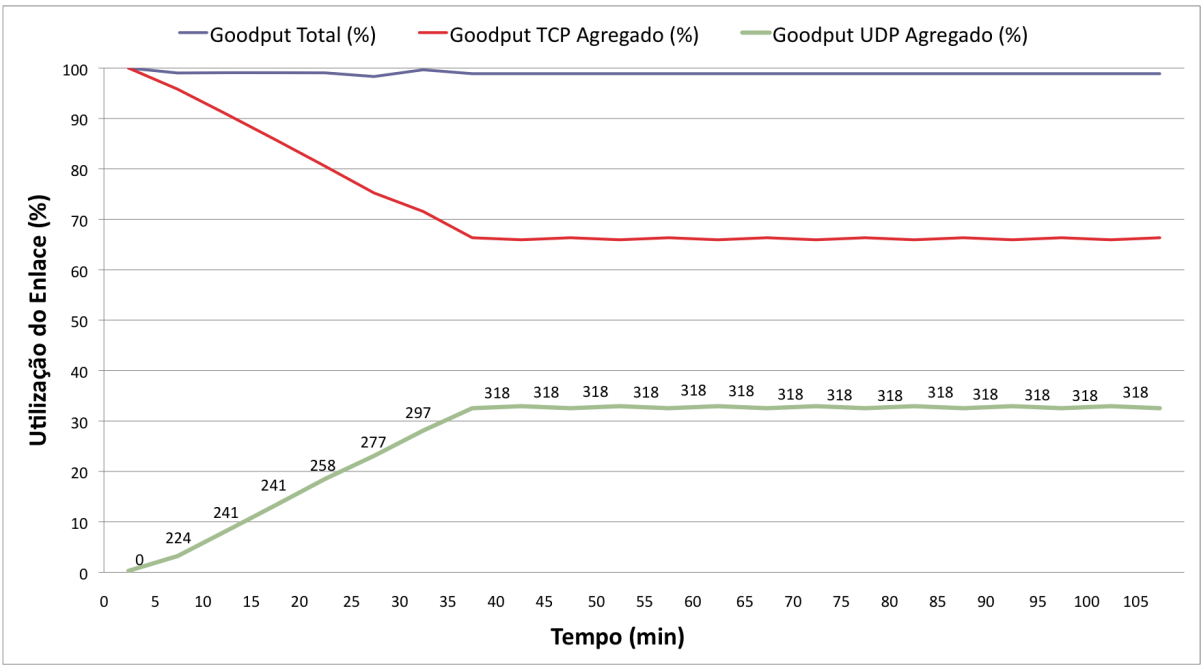

FIG. 6.2: Utilização do Enlace com Tarifa Máxima Aceitável pelos Clientes.

feitas através dos enlaces que interligam os roteadores Creta, Cyprus e Thassos.

A metodologia utilizada no terceiro teste é a mesma que foi utilizada no primeiro teste deste capítulo. Pedidos de transmissão UDP são feitos a cada 5 minutos apenas durante os primeiros 90 minutos e depois não mais. Cada pedido solicita a alocação de 50 kbits/s por um período de 90 minutos.

Os resultados obtidos são mostrados na FIG. 6.4. Nela estão os dados relativos aos dois enlaces utilizados na transmissão. O enlace entitulado "Enlace 1" é referente ao que conecta os roteadores Creta e Cyprus. Já o "Enlace 2" conecta os roteadores Cyprus e Thassos. As linhas vermelha e azul representam o *goodput* percentual em relacão à capacidade dos respectivos enlaces obtido pelo tráfego TCP agregado. As linhas roxa e marrom representam o *goodput* total de cada enlace, ou seja, a soma do *goodput* do tráfego UDP agregado e do tráfego TCP agregado respectivos a cada enlace. Observando essas duas linhas é fácil perceber que a capacidade do enlace foi bem aproveitada. A média de utilização total do "Enlace 1" foi de 98% e a do "Enlace 2" foi de 99%, tendo como menores índices 94% e 91%, respectivamente.

Já as linhas verde e laranja da FIG. 6.4 possuem duplo significado. Além de indicarem os respectivos *goodput* percentuais do tráfego UDP em relação às capacidades dos enlaces, junto delas também está indicado a tarifa vigente para acomodar no percurso uma nova

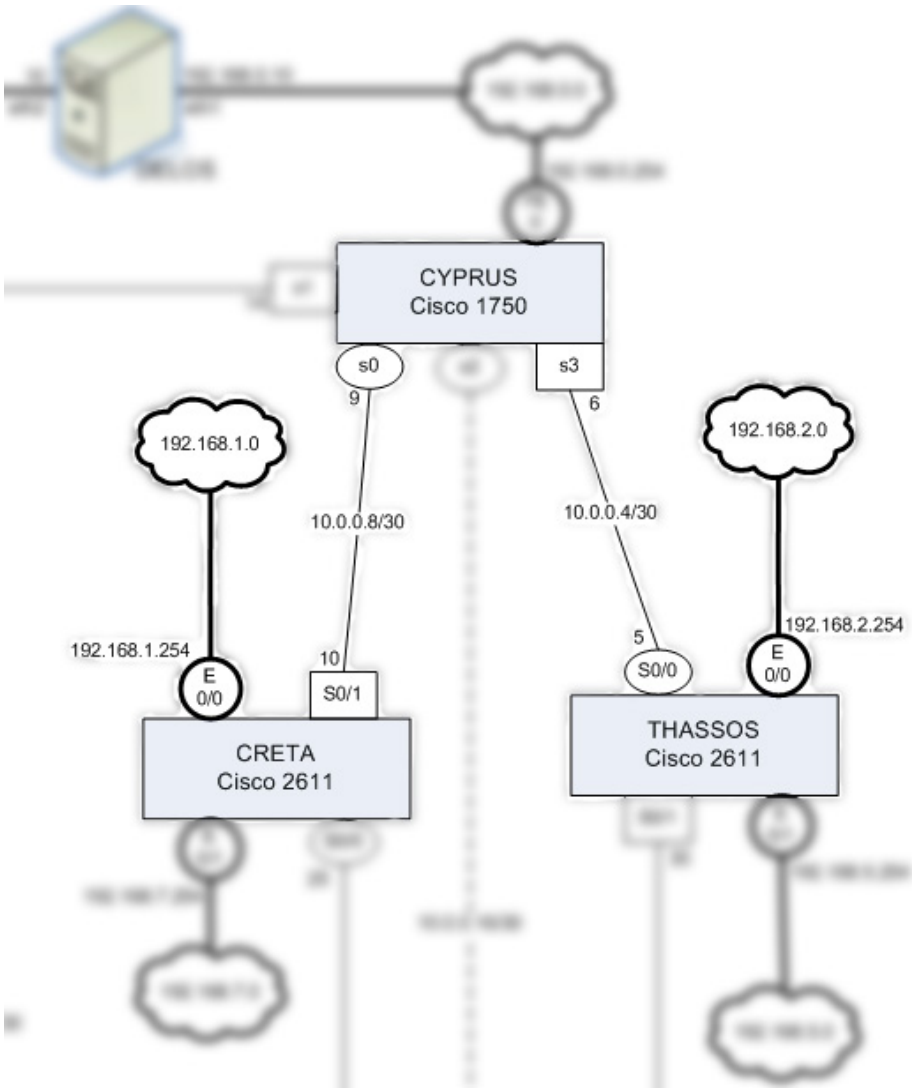

FIG. 6.3: Segundo Cenário de Teste.

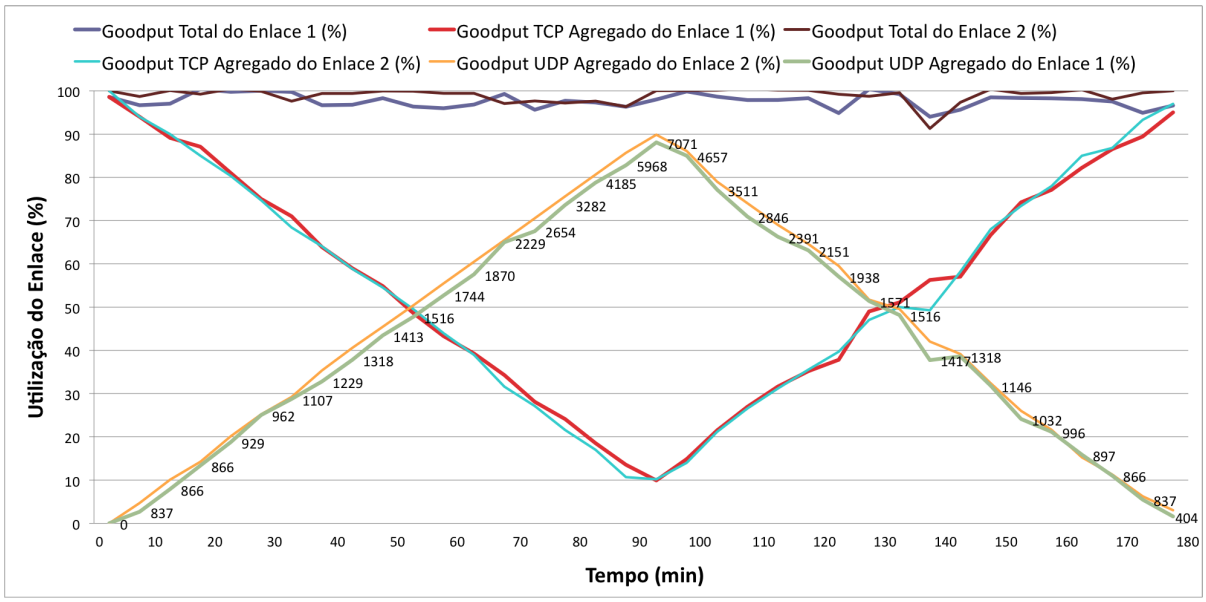

FIG. 6.4: Utilização do Enlace com Tarifa Máxima Aceitável pelos Clientes.

transmissão padrão de 50 kbits/s durante 90 minutos utilizando o protocolo UDP. Quanto menor o rendimento do tráfego TCP causado pela maior utilização de tráfego UDP, maior é a tarifa vigente.

Como durante os primeiros 90 minutos do teste novos pedidos de transmissão UDP são feitos, a tarifa e o *goodput* do tráfego UDP sempre aumentam no decorrer do tempo. Após os 90 minutos iniciais, as transmissões passam a terminar, o que diminui o *goodput* do tráfego UDP e, consequentemente, aumenta o *goodput* do tráfego TCP e diminui a tarifa.

Como o percurso é maior em relação ao do primeiro teste, as tarifas vigentes também são maiores e aproximadamente o dobro. Quando o tráfego UDP ocupa 55% dos enlaces, a tarifa (1744 u.m.) é mais do que o dobro do que quando a ocupação é de 15% (866 u.m.). No caso em que o tráfego UDP ocupa 90% dos enlaces, a tarifa (7.071 u.m.) é mais de 8 vezes maior do que quando a ocupação é de 15%. A diferença em valores absolutos entre as tarifas neste último caso é de 6.205 u.m.

Fica demonstrado que o quinto e último objetivo dos testes foi alcançado.

7 CONSIDERAÇÕES FINAIS

Os testes realizados confirmam que a arquitetura aqui proposta safisfaz seus objetivos, sendo capaz de evitar o congestionamento do tráfego TCP, gerar garantias ao tráfego UDP agregado, desestimular a entrada de novos fluxos UDP sem baní-los da rede quando o tráfego TCP estiver experimentando baixo desempenho e permitir ao administrador da rede fixar um percentual de utilização do enlace para tráfego UDP através do controle de tarifa.

A arquitetura proposta é justa, pois a tarifa cobrada do usuário que deseja transmitir tráfego UDP é sempre maior quanto maior for o prejuízo que este causar ao tráfego TCP.

A solução desenvolvida é híbrida, pois utiliza técnicas propostas em soluções baseadas em Rede e em Preço.

Apesar de diversos autores já terem analisado o problema abordado nesta dissertação, o presente trabalho possui outros dois grandes diferenciais.

O primeiro é o fato dos resultados obtidos serem fruto de experimentos reais e não simulações. Entre todos os estudos sobre o problema de coexistência de fluxos TCP e UDP e propostas de solução analisados pelo presente trabalho, nenhum foi testado pelos seus respectivos autores em ambientes reais, utilizando ativos de rede reais. Nesta dissertação, todos os experimentos, não só os testes da implementação da arquitetura, mas também aqueles que expõem o problema, foram realizados em ambiente real.

O segundo diferencial é o desenvolvimento de um produto, *software*, ainda em estágio inicial, mas capaz de atuar sobre o problema abordado.

Porém, é importante destacar uma condição necessária para que o sistema aqui implementado funcione como esperado. A desutilidade do tráfego TCP é obtida através da análise de seu desempenho. Esta métrica está relacionada com a capacidade de transferir dados por unidade de tempo. A questão é que este tipo de análise não alcança os tráfegos transacionais, de curta duração e pouco volume de dados. Esse tipo de tráfego, denominado por diversos autores como “Tráfego Rato”, se contrapõe ao “Tráfego Elefante”, cuja característica é ter longa duração e alto volume de dados (FIOREZE, 2007) (MORI, 2004) (MARSAN, 2005). Este tipo sim, o tráfego elefante, é alcançado pela arquitetura aqui proposta.

Com relação à forma de tarifação, a arquitetura é robusta o suficiente para acomodar outras formas de tarifação baseadas em informações de fluxos. Pequenas modificações no módulo "Atualiza Tarifa" (vide FIG. 5.2) permitem uma extração diferenciada de informações do banco de dados e até mesmo alteração na função de cálculo da tarifa. Já para as funções de tarifação que utilizem parâmetros como RTT ou *jitter,* é necessário estender a arquitetura com um novo módulo que verifique na rede de dados administrada o valor do parâmetro a ser utilizado.

Algumas particularizações na implementação da arquitetura tiveram que ser feitas para adequá-la à disponibilidade de equipamentos do Laboratório de Redes onde os testes foram conduzidos.

Uma delas é o fato da implementação da arquitetura funcionar apenas com roteadores Cisco. Esta simplificação foi feita porque apenas essa marca de roteador estava disponível.

A outra particularização é a utilização de *shell scripts* para realizar a tarefa de configuração dos roteadores da rede ao invés da uma tecnologia mais robusta, popular e escalável. A opção por utilizar SNMP não pode ser feita devido à ausencia das MIBs necessárias nos roteadores Cisco do Laborátorio de Redes de Dados. Para utilizar SNMP na implementação, são necessárias modificações nos módulos "*Define e Armazena Comandos*" e "*Reconfigura Cenário*".

A solução aqui proposta destina-se a tratar o problema da coexistência de fluxos TCP e UDP dentro de um ambiente controlado. Para que a arquitetura seja extendida aos ambientes vizinhos, é necessário alguma forma de cooperação através de acordos entre os Bandwidth Brokers de cada ambiente controlado.

7.1 TRABALHOS FUTUROS

A arquitetura proposta pode identificar de maneira equívoca a desutilidade sentida pelo tráfego TCP quando este for composto apenas por fluxos transacionais. Este erro pode gerar uma tarifa mais alta do que a correta. Para evitar esse problema, além das duas filas gerenciadas pelo escalonador *Custom Queueing*, uma terceira fila deve ser criada. Esta fila agregaria todas as conexões TCP do tipo "rato" e receberia, através do escalonador *CQ*, um percentual fixo de utilização dos enlaces. A identificação desse tipo de tráfego pode ser feita através de uma análise mais aprofundada dos fluxos da rede. Através da 5-tupla ip origem/destino, porta origem/destino e protocolo, cada fluxo deve ser analisado com relação ao tempo de transmissão e o volume de dados enviados e, então, identificado

como “rato” ou “elefante” . O problema desta solução é a escalabilidade em redes com tráfego intenso, como em sistemas autônomos.

Outra funcionalidade que deve ser implementada pelo *software* gerenciador para que a arquitetura proposta possa ser utilizada de fato como um produto é a criação de regras nos roteadores de borda com os clientes que limitem o *throughtput* da transmissão UDP solicitada. Atualmente, o controle é feito apenas sobre o tráfego UDP agregado, o que permite que um cliente UDP utilize uma fatia do percentual do enlace contratado por outra transmissão UDP.

O roteamento na arquitetura proposta é feito pelos roteadores utilizando apenas o protocolo OSPF. Em alternativa, o roteamento poderia ser feito utilizando a tecnologia de múltiplas topologias (CISCO, 2007) (KVALBEIN, 2007), onde o estado (tarifa) do enlaces seria utilizado para determinar a métrica de roteamento da topologia dedicada ao tráfego UDP.

Com relação à reserva de recursos nos enlaces, deve ser expandido o módulo de gerenciamento de recursos para comportar reservas diferentes nas interfaces de um mesmo roteador. Para tanto, basta criar uma *queue-list*, cada uma com as filas UDP e padrão (ver apêndice 9.3), para cada interface, e gerenciá-las individualmente.

O módulo “*Define e Armazena Comandos*” deve ser extendido para permitir a configuração de diferentes tipos de roteadores. A implementação da arquitetura contempla apenas equipamentos da marca Cisco. Além da alteração no módulo citado, é necessário criar uma tabela no banco de dados para armazenar os modelos dos roteadores utilizados no sistema.

Ainda são necessários mais testes que comprovem a escalabilidade da arquitetura em situações onde a quantidade de fluxos UDP seja de ordem maior do que a testada. O Bandwidth Broker obtém as informações de estado da rede através da leitura no banco de dados de cada fluxo UDP, porém a análise e o tratamento é feito com relação ao tráfego agregado. Já nos roteadores, a reserva de recursos também é feita para tráfego UDP agregado, somente as regras de desbloqueio de tráfego UDP é que são individuais para cada fluxo. Fazendo uma comparação com as arquiteturas Intserv e Diffserv, pode-se dizer que a obtenção da informação sobre o estado dos enlaces é feita como na arquitetura Intserv, mas o tratamento dos dados e a reserva de recursos é realizado como na Diffserv. Por esses motivos, acredita-se que a arquitetura proposta seja escalável.

8 REFERÊNCIAS BIBLIOGRÁFICAS

ADVENTNET, I. Manageengine netflow analyzer. URL `http://manageengine.adventnet.com/products/netflow/index.html`. *Visitado em 25 de setembro de 2008.*

ALBUQUERQUE, C., VICKERS, B. J. e SUDA, T. Network border patrol - preventing congestion collapse and promoting fairness in the internet. págs. 173–186. **IEEE/ACM Transactions On Networking**, Vol. 12, No 1, 2004.

BALLIACHE, L. Network qos using cisco howto. 2003. URL `http://www.opalsoft.net/qos/WhyQoS.htm`. *Visitado em 25 de setembro de 2008.*

BRISCOE, B. Flow rate fairness: Dismantling a religion. págs. 65–74. **ACM SIGCOMM Computer Communication Review**, Vol. 37, No. 2, 2007.

BYUN, J. e CHATTERJEE, S. A strategic pricing for quality of service (qos) network business. págs. 2561–2572. **Proceedings of the Tenth Americas Conference on Information Systems**, 2004.

CHODOREK, R. R. Some aspects of rtp - tcp coexistence in universal multiservice networks. **IEEE**, 2000.

CISCO. Cisco ios netflow configuration guide. **Cisco Systems, Inc.**, 2006.

CISCO. Multi-topology routing. **Cisco Systems, Inc.**, 2007.

COCCHI, R., SHENKER, S., ESTRIN, D. e ZHANG, L. Pricing in computer networks - motivation, formulation, and example. págs. 614–627. **IEEE/ACM Transactions On Networking**, Vol. 1, No 6, 1993.

COMSCORE, I. Global internet information provider. 2008. URL `http://www.comscore.com/press/release.asp?press=2444`. *Visitado em 16 de outubro de 2008.*

COTER. Principais aspectos do sistema de comando e controle do exército e da força terrestre. **Ministério da Defesa - Exército Brasileiro - Comando de Operações Terrestres**, 2002.

FIOREZE, T., WOLBERS, M. O., VAN DE MEENT, R. e PRAS, A. Finding elephant flows for optical networks. págs. 627–640. **10th IFIP/IEEE International Symposium on Integrated Network Management**, 2007.

FLOYD, S. e FALL, K. Promoting the use of end-to-end congestion control in the internet. págs. 458–472. **IEEE/ACM Transactions On Networking**, Vol 7, No 4, 1999.

GAO, X. e SCHULMAN, L. J. Feedback control for router congestion resolution. **PODC'05**, 2005.

KESHAV, S. *An Engineering Approach to Computer Networking*, capítulo Scheduling, págs. 209–263. **Addison-Wesley**, 2001.

KEY, P., MASSOULIÉ, L., BAIN, A. e KELLY, F. Fair internet traffic integration: Network flow models and analysis. págs. 1338–1352. **Annales des Telecommunications**, No. 59, 2004.

KOHLER, E., HANDLEY, M. e FLOYD, S. Designing dccp: Congestion control without reliability. **SIGCOMM**, 2006.

KUROSE, J. F. e ROSS, K. W. *Computer Networking: A Top-down Approach Featuring the Internet*. **Addison-Wesley**, 2004.

KVALBEIN, A. e LYSNE, O. How can multi-topology routing be used for intradomain traffic engineering. págs. 280–284. **SIGCOMM Workshop on Internet Network Management**, 2007.

LAI, Y.-C. e CHIEN, S.-C. A tcp-friendly congestion control to guarantee smoothness by slack term. **IEEE**, 2004.

LEE, C., LEE, D., KOO, J., BANERJEE, S. e CHUNGA, J. Cli-based tcp: an end-to-end proactive approach robust to traffic load. **Elsevier**, 2008.

LEFELHOCZ, C., LYLES, B., SHENKER, S. e ZHANG, L. Congestion control for best-effort service - why we need a new paradigm. págs. 10–19. **lEEE Network** - January-iFebruary, 1996.

MACKIE-MASON, J. K. e VARIAN, H. R. *Public access to the Internet*, capítulo Pricing the Internet, págs. 269–314. **MIT Press**, 1995.

MANI, P. e PETR, D. W. Investment function - enhanced fairness and performance in multi-hop wireless networks. **IEEE**, 2006.

MARSAN, M. A., GARETTO, M., GIACCONE, P., LEONARDI, E., SCHIATTARELLA, E. e TARELLO, A. Using partial differential equations to model tcp mice and elephants in large ip networks. págs. 1289–1301. **IEEE/ACM Transactions on Networking**, Vol. 13, No. 6, 2005.

MAURO, D. e SCHMIDT, K. *Essential SNMP*. **O'Reilly**, 2001.

MICHEL, N. F. e FONSECA, N. L. S. Uma investigação sobre a escalabilidade de variantes do protocolo tcp para redes de alta velocidade. págs. 635–652. **Anais do XXVII Congresso da SBC**, 2007.

MILLER, D. Flowd. 2008. URL `http://www.mindrot.org/projects/flowd/`. *Visitado em 25 de setembro de 2008.*

MORI, T., KAWAHARA, R., NAITO, S. e GOTO, S. On the characteristics of internet traffic variability: spikes and elephants. págs. 99–106. **International Symposium on Applications and the Internet**, 2004.

NAGLE, J. Congestion control in ip/tcp internetworks. págs. 11–17. **Ford Aerospace and Communications Corporation**, 1984.

NETRATINGS, N. Nielsen online. 2008. URL `http://www.nielsen-netratings.com`. *Visitado em 16 de outubro de 2008.*

NICHOLS, K., JACOBSON, V. e ZHANG, L. A two-bit differentiated services architecture for the internet. 1999.

OTT, T. J., LAKSHMAN, T. e WONG, L. H. Sred: Stabilized red. **IEEE**, 1999.

PERSICO, G. Neye. 2005. URL `http://neye.unsupported.info`. *Visitado em 25 de setembro de 2008.*

RIBEIRO, R. D. e SALLES, R. M. Mecanismos para controle de tráfego udp através de política de preços baseada na utilidade. **XXV Simpósio Brasileiro de Telecomunicações**, 2007.

RICHARDS, A., ROGERS, G., ANTONIADES, M. e WITANA, V. Mapping user level qos from a single parameter. **CSIRO e Co-operative Research Center Program of Department of Industry of the Government of Australia**, 1999.

ROBERTS, J. W. Internet traffic, qos and pricing. págs. 1389–1399. **Proceedings Of The IEEE**, Vol. 92, No. 9, 2004.

SHENKER, S., CLARK, D., ESTRIN, D. e HERZOG, S. Pricing in computer networks - reshaping the research agenda. págs. 183–201. **Elsevier Science Telecommunications Policy**, Vol. 20, No. 3, 1996.

SHENKER, S. Fundamental design issues for the future internet. págs. 1176–1188. **IEEE Journal On Selected Areas In Communications**, Vol. 13, No. 7, 1995.

STALLINGS, W. *Data Computer Communications*. **Prentice Hall**, 2000.

STATS, I. W. Internet usage statistics - the internet big picture. 2008. URL `http://www.internetworldstats.com/stats.htm`. *Visitado em 16 de outubro de 2008.*

STIGLER, G. J. *Análise Microeconômica - Teoria dos Preços*. **Atlas**, 1970.

SUNMICROSYSTEMS. Mysql. 2007. URL `http://www.mysql.com/`. *Visitado em 25 de setembro de 2008.*

TANG, A., WANG, J. e LOW, S. H. Understanding choke - throughput and spatial characteristics. págs. 694–707. **IEEE/ACM Transactions On Networking**, Vol. 12, No 4, 2004.

VAN, N. H., POPOV, O. e POPOVA, I. Combined model for congestion control. **28th Int. Conf. Information Technology Interfaces**, 2006.

VARIAN, H. R. *Intermediate Microeconomics: A Modern Approach*. **W. W. Norton & Company**, 2006.

VILLANUSTRE, F. Flow loader & virtual information output. 2002. URL `http://flavio.sourceforge.net/`. *Visitado em 25 de setembro de 2008*.

WANG, J., TANG, A. e LOW, S. H. Maximum and asymptotic udp throughput under choke. **SIGMETRICS'03**, 2003.

WANG, X. e SCHULZRINNE, H. Pricing network resources for adaptive applications. págs. 506–519. **IEEE/ACM Transactions On Networking**, Vol. 14, No 3, 2006.

XU, Y. e GUÉRIN, R. Individual qos versus aggregate qos: A loss performance study. págs. 370–383. **IEEE/ACM Transactions On Networking**, Vol. 13, No 2, 2005.

YILMAZ, S. e MATTA, I. On class-based isolation of udp, short-lived and long-lived tcp flows. Technical report, Boston, MA, USA, 2001.

9 APÊNDICES

9.1 BANDWIDTH BROKER

9.1.1 GERENCIADOR

Código fonte do *software* no cd em anexo. Para compilar, segue um exemplo:

```
gcc -lmysqlclient -o gerenciador bb.c
```

9.1.2 COLETOR DE FLUXOS

Código fonte da versão alterada do *software* NEye (PERSICO, 2005) no cd em anexo. Os seguintes arquivos originais foram alterados:

```
/neye-1.0.1/config.h
/neye-1.0.1/include/structs.h
/neye-1.0.1/neyed/neyed.h
/neye-1.0.1/neyed/parser.c
/neye-1.0.1/neyed/storage.c
/neye-1.0.1/neyed/mysql/createdb
/neye-1.0.1/trafgen/trafgen.c
```

Segue o modo de instalação do *software* coletor de fluxos.

1) Instalar o banco de dados MySQL e criar o usuário root com senha em branco

2) Iniciar o MySQL

3) Instalar o SQLITE e o SQLITE3

4) Criar a base de dados FLOWDB e as tabelas FLOWS e ENLACES conforme script createdb

5) Povoar a tabela ENLACES com os dados relativos a rede de dados administrada

6) Instalar a versão modificada do NEye fornecida conforme os comandos a seguir alterando os diretórios conforme a necessidade:

a) ./makedist

b) ./configure -with-telcosux -with-mysql-include=/usr/include/mysql -with-mysql-libraries=/usr/lib/mysql -with-sqlite-include=/usr/local/include -with-sqlite-libraries=/usr/local/lib

c) make

d) make install

7) Iniciar o NEye utilizando o IP do host coletor de fluxos

/opt/neye/sbin/neyed -m *IP*

9.1.3 SCRIPT

Copiar o arquivo `configura.sh` em anexo no cd para o mesmo diretório do arquivo executável do *software* Gerenciador.

```
#/bin/bash!

# tamanho medio dos pacotes
tamPacote=1250

# inicializa as variaveis com os valores percentuais recebidos
ip=$1
QoS=$2

# inicializa cada variavel de senha com seu valor padrao
pw="senha"
pwEn="senhaModoPrivilegiado"

# calcula valor das variaveis do escalonador CQ
let "filaUDP=$tamPacote*$QoS/10"
let "filaDefault=$tamPacote*(100-$QoS)/10"

# executa os comandos em roteador Cisco
(echo
sleep 2
echo $pw
sleep 2
```

```
echo en
sleep 2
echo $pwEN
sleep 2
echo configure terminal
sleep 2
echo queue-list 1 queue 1 byte-count $filaUDP
sleep 2
echo queue-list 1 queue 2 byte-count $filaDefault
sleep 2
echo exit
sleep 2
echo exit) | telnet $ip 2>/dev/null
```

9.2 CLIENTE

Código fonte do *software* no cd em anexo.

9.3 CONFIGURANDO OS ROTEADORES

Para o correto funcionamento da arquitetura proposta nesta dissertação é necessário que os roteadores da rede administrada sejam previamente configurados para utilizar o escalonador *Custom Queueing* e estejam habilitados à exportar, através da tecnologia de fluxos, as informações pertinentes à utilização das interfaces. Segue um exemplo de configuração para o caso de utilização de roteadores Cisco.

1) Logar em cada roteador em modo privilegiado

2) Criar a lista de acesso para a fila UDP do escalonador CQ

```
access-list 101 permit udp any any
```

3) Criar a fila UDP do escalonador CQ

```
queue-list 1 protocol ip 1 list 101
```

4) Criar a fila padrão para todos os outros protocolos

```
queue-list 1 default 2
```

5) Reservar 1% dos enlaces com os comandos

```
queue-list 1 queue 1 byte-count 125
queue-list 1 queue 2 byte-count 12375
```

6) Em cada interface do roteador habilitar o escalonador CQ

```
custom-queue-list 1
```

7) Habilitar o NetFlow para que seja exportado através de uma interface

```
ip flow-export source INTERFACE
```

8) Escolher a versão correta do NetFlow

```
ip flow-export version 5
```

9) Escolher o IP do Coletor de Fluxos e a porta correta utilizada por este

```
ip flow-export destination IP 2055
```

10) Habilitar o Cisco Express Forwarding no roteador para otimizar o desempenho

```
ip cef
```

11) Escolher o período de 1 minuto para que o roteador consolide as informações de tráfego das interfaces

```
ip flow-cache timeout active 1
```

Livros Grátis

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

Milhares de Livros para Download:

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