Carlos Benjamin Pazzianotto

“Implementação de controle de banda com *Packet Filter* - PF e *Alternate Queueing* - ALTQ em FreeBSD”

Monografia de Pós-Graduação “*Lato Sensu*” apresentada ao Departamento de Ciência da Computação para obtenção do título de Especialista em “Administração em Redes Linux”

Orientador
Prof. Sandro Melo

Lavras
Minas Gerais - Brasil
2006

Carlos Benjamin Pazzianotto

“Implementação de controle de banda com *Packet Filter* - PF e *Alternate Queueing* - ALTQ em FreeBSD”

Monografia de Pós-Graduação “*Lato Sensu*” apresentada ao Departamento de Ciência da Computação para obtenção do título de Especialista em “Administração em Redes Linux”

Aprovada em *28 de setembro de 2006*

Prof. Joaquim Quinteiro Uchôa

Prof. Denilson Vedoveto Martins

Prof. Sandro Melo
(Orientador)

Lavras
Minas Gerais - Brasil

Agradecimentos

Agradeço a minha esposa Leonice pela paciência durante o curso, aos meus filhos Mateus e Marília por *liberarem o computador* para as minhas tarefas, aos colegas Ana Paula, Carlinhos, Cláudia, Maritê por *segurarem a barra* no serviço na minha ausência, ao Victor pela ajuda nas referências bibliográficas, ao João Marques pela ajuda na estruturação deste trabalho, a minha chefe Maria Amélia pelo apoio, ao Prof. Joaquim pela ajuda nas *encruzilhadas* do curso e pelo incentivo em usar este excelente LaTeX que tanto facilitou no trabalho e ao meu orientador Prof. Sandro Melo, pelas *doses* freqüentes de otimismo e paciência na condução do trabalho.

Sumário

1 Introdução **1**

1.1 O crescente uso das redes de computadores 1
1.2 Justificativa 3
1.3 Objetivos 4
1.3.1 Objetivo Geral 4
1.3.2 Objetivos específicos 4
1.4 Motivação e escopo 4
1.5 Estado da arte 4
1.5.1 O *Dummynet* 5
1.5.2 O *Traffic Control - TC* 6
1.5.3 O *M0n0wall* 6
1.5.4 O *Pfsense* 6
1.5.5 Soluções proprietárias 6
1.6 Organização do trabalho 7

2 O *Alternate Queueing - ALTQ* **9**

2.1 Histórico 9
2.2 Objetivos 9
2.3 Desenho 10
2.4 Integração do *Alternate Queueing - ALTQ* com o *Packet Filter - PF* 10
2.5 O *Alternate Queueing - ALTQ* e as disciplinas de filas 11

3 As Filas **13**

3.1 Fundamentos de filas 13
3.2 Componentes de Filas 13
3.3 Disciplinas de filas 15
3.3.1 *FIFO - First in First out* 15
3.3.2 Filas baseadas em classes - *CBQ - Class Based Queueing* . 16
3.3.3 Filas com prioridades - *PQ - Priority Queueing* 18

3.3.4 Filas Justas *WFQ - Weighted Fair Queueing* 19
3.3.5 Outras disciplinas de filas 20
3.4 Algoritmos de controles de congestionamentos 20
3.4.1 ECN - Explicit Congestion Notification 20
3.4.2 *RED - Randon Early Detection* 21

4 O *Configuração do Alternate Queueing - ALTQ* **23**
4.1 O *Alternate Queueing - ALTQ* no *FreeBSD* 23
4.2 Parâmetros de operação do *Alternate Queueing - ALTQ* 23
4.2.1 Diretiva *altq* 24
4.2.2 Diretiva *queue* 25
4.2.3 Atribuição de tráfego às filas 25

5 Materiais e Métodos **29**
5.1 Implementação de controle de banda com *Alternate Queueing - ALTQ* 29
5.2 Ambiente Operacional 29
5.3 Arquivo de regras 29
5.4 Parâmetros e regras para avaliação de efetividade 30
5.5 Forma de obtenção dos dados 30
5.5.1 Obtenção dos dados de filas 30
5.5.2 Obtenção da vazão entre as máquinas *onix* e a *diamante* . 31
5.5.3 Obtenção de dados da fila q_tcp 31
5.5.4 Obtenção de dados da fila q_ssh 32
5.5.5 Interceptação das filas *TCP* e *SSH* 33
5.6 Limitações do experimento 33

6 Resultados obtidos **35**
6.1 Dados obtidos 35
6.1.1 Dados da vazão (*throughput*) *onix-diamante* 35
6.1.2 Dados obtidos pela geração de tráfego *SSH* 35
6.1.3 Dados obtidos pela geração de tráfego *TCP* através do software *Netperf* 35
6.2 Análise dos dados obtidos 35
6.2.1 Tráfego *TCP* 35
6.2.2 Tráfego *SSH* 36

7 Conclusão **39**
7.1 Possibilidades para trabalhos futuros sobre o tema 39

A Arquivos de configuração e scripts **43**
A.1 /etc/pf.conf 43
A.2 Scripts 44
A.2.1 Estimativa de vazão (*throughput*) 44
A.2.2 Gerador de tráfego TCP 44
A.3 Aplicações das diretrizes 44

Lista de Figuras

1.1 Usuários de Internet em cada 100 habitantes no Brasil 2
1.2 (CORREIA; SILVA, 2002) Congestionamento em função de aumento de tráfego de pacotes 3

2.1 Detalhe da alteração do *Alternate Queueing - ALTQ* no Kernel . . 11

3.1 Visão geral de uma arquitetura de filas 14
3.2 Disciplina de filas *FIFO -First in First out* 16
3.3 Disciplina de filas *CBQ - Class Based Queueing* 17
3.4 Disciplina de filas *PQ - Priority Queueing* 18
3.5 Disciplina de filas *WFQ - Weighted Fair Queueing* 19

5.1 Ambiente operacional 30
5.2 Forma de obtenção dos dados 32

6.1 Comportamento do tráfego *TCP* 37
6.2 Comportamento do tráfego *SSH* 38

A.1 Esquema da rede 45

Lista de Tabelas

3.1 Estrutura de filas/classes na disciplina *CBQ - Class Based Queueing* 17
3.2 Estrutura de filas/classes na disciplina *CBQ - Class Based Queueing* - prioridades 18
3.3 Estrutura de filas/classes na disciplina *PQ - Priority Queueing* . . 19

6.1 *Throughput onix-diamante* 36
6.2 Geração de tráfego *SSH* através do *sft* 37
6.3 Geração de tráfego *TCP* através do *Netperf* 38

Capítulo 1

Introdução

1.1 O crescente uso das redes de computadores

A transformação de informações - números, gráficos, textos, fotos, voz, imagens, dentre outras - para o formato digital e a contínua queda verificada nos preços de computadores, tornou o uso destas máquinas, conectadas em rede, dramaticamente crescente, em nível mundial, quer no ambiente de negócios quer no ambiente acadêmico e mesmo doméstico.

Estes fatores aliados à necessidade de rapidez no processo de trocas de informações, essencial neste mundo globalizado, causaram um espantoso aumento no uso de redes de computadores. Em conseqüência, e como exemplo significativo desta evolução nos meios e formas de comunicação tem-se a rede mundial, a Internet, uma rede singular por sua abrangência. Segundo a (ONU, 2005), de cada 100 pessoas, 0,01 (1 em 10.000) usavam a internet no Brasil em 1992, em 2002 esta proporção subiu para 8,22 pessoas em cada 100 (Figura 1.1)[1] O ocorrido no Brasil pode-se, guardando-se as devidas proporções, ser generalizado para o mundo todo com destaque, principalmente, para os países ocidentais.

Segundo (TAKAHASHI, 2000), a universalização dos serviços de informação e comunicação, o desenvolvimento de novas aplicações como telemedicina, ensino à distância, comércio eletrônico indicam o uso potencial a ser explorado com maior intensidade, tendendo, desta forma, a gerar crescimentos exponenciais do uso da internet.

Tal como nos sistemas viários das grandes metrópoles com seus freqüentes problemas de trânsito em função do aumento de tráfego dos meios de locomoção,

[1] Conforme (STATS, 2006) a proporção de uso da internet no Brasil para o ano de 2005 é de 14,1 para cada 100 pessoas com um aumento de cerca de 400% em relação ao ano 2000. A mesma fonte informa que o país é o décimo colocado entre os que mais utilizam a rede internet.

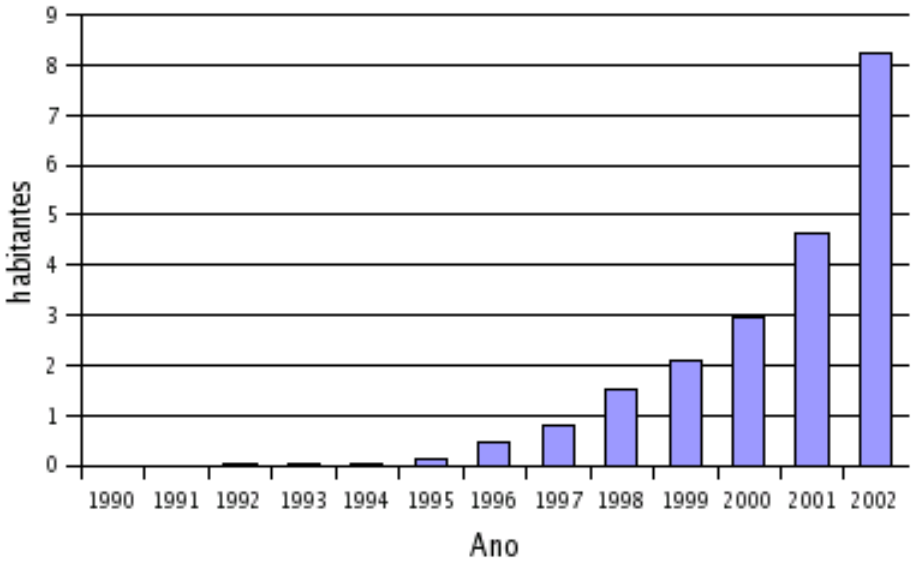

Figura 1.1: Usuários de Internet em cada 100 habitantes no Brasil

o inevitável e crescente grau de utilização da rede também tende a gerar congestionamentos.

Quando há *pacotes*[2] demais presentes na rede, o seu desempenho diminui. Essa situação é chamada de congestionamento. Assim, se o número de pacotes depositados pelos *hosts*[3] é compatível com a capacidade de transporte da rede, eles são entregues (exceto aqueles com erros de transmissão), e o número entregue é proporcional ao número enviado. Entretanto, quando existe aumento de tráfego, os *roteadores*[4] já não são capazes de suportá-lo e deixam de entregar os pacotes (Figura 1.2). Em situações de tráfego pesado o desempenho diminui muito e quase nenhum pacote é entregue (CORREIA; SILVA, 2002).

Compete ao administrador de redes intervir de maneira a minimizar ou evitar os problemas de congestionamentos. Estas intervenções podem se verificar na formulação de *políticas de uso da informação* consistentes e na observação do seu cumprimento, na vigilância dos ativos de redes elaborando estratégias de troca de equipamentos defasados ou mal dimensionados, na pesquisa de ferramentas de gerência e na administração de controle de tráfego.

[2] *pacotes* - é a informação seccionada juntamente com elementos de controle que foram enviados por uma fonte serão reconstruídos e reordenados no destino

[3] *hosts* - são computadores pertencentes à rede

[4] *roteador* - dispositivo que interliga as redes

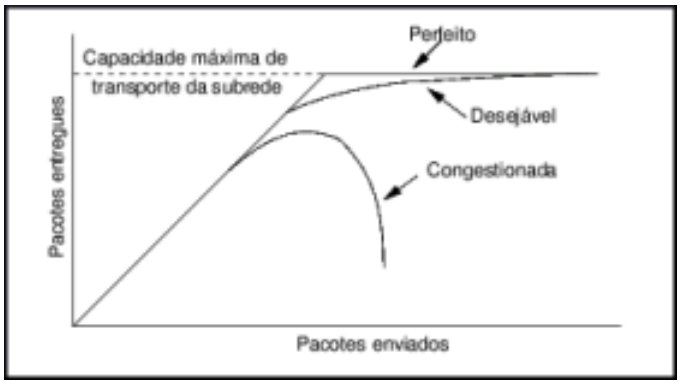

Figura 1.2: (CORREIA; SILVA, 2002) Congestionamento em função de aumento de tráfego de pacotes

1.2 Justificativa

Em vista da crescente necessidade de se evitar o congestionamento causado pelo também crescente tráfego nas redes de computadores no presente e contribuir com maior eficiencia nas trocas de informações, várias possibilidades tem sido indicadas como possível solução do problema.

Segundo (CHO, 1999), existe uma grande demanda e expectativas por gerenciamento de tráfego de rede. Embora tenha-se variedade de tecnologias, não existe uma que tenha abrangência total. Torna-se importante o conhecimento das vantagens e limitações das diferentes tecnologias, assim como os problemas de tráfego nas diversas redes. Em redes cuja diversidade abranja de lentos modems à rápidos cabos óticos, os problemas de tráfego não são fáceis de serem isolados. O conhecimento de disciplinas de filas tem ajudado na obtenção de resultados, porém a interação com outros mecanismos como configurações de sistemas e serviços não são ainda definitivos e ainda necessitam de mais estudo e experiência.

Várias tentativas de elaboração de soluções para controle de tráfego nas redes de computadores tem sido desenvolvidas, dentre estas inclue-se o *software Alternate Queueing - ALTQ*. Conforme (CHO, 1999), com o uso do *software Alternate Queueing - ALTQ* pode-se intervir no tráfego de saída de um computador através da manipulação de filas, impondo limites e priorizando o tráfego.

1.3 Objetivos

1.3.1 Objetivo Geral

Em termos gerais, pode-se definir o objetivo deste trabalho como explorar as possibilidades do uso do *software Alternate Queueing - ALTQ* no controle do uso de bandas para evitar congestionamentos e efetuar de forma suscinta a descrição de sua base teórica e divulgar os resultados para os técnicos de administração de redes da Embrapa - Empresa Brasileira de Pesquisa Agropecuária.

1.3.2 Objetivos específicos

Testar através de experimento e avaliar o uso do *software Alternate Queueing - ALTQ*, no ambiente Embrapa, como ferramenta que permita o controle do uso de bandas[5] de redes de computadores.

1.4 Motivação e escopo

Este trabalho aborda o uso da ferramenta *Alternate Queueing - ALTQ* como uma possibilidade para de controle de bandas de redes, tendo em vista os processos de congestionamentos que podem se tornar freqüentes, em virtude não só da intensificação mas também das novas possibilidades de uso que estão surgindo para as redes de computadores. Pretende-se testar e implementar na Embrapa Meio Ambiente unidade temática da Embrapa - Empresa Brasileira de Pesquisa Agropecuária, que segue diretriz do governo federal para migração de seus sistemas operacionais para *software livre*, e que adotou o sistema operacional *FreeBSD*[6] em suas máquinas servidoras. Objetiva-se também divulgar os resultados deste trabalho para os técnicos das demais unidades da Embrapa, mais de quarenta espalhadas pelo país, de forma a auxiliá-los com conhecimentos e ferramenta para controle de banda de redes.

1.5 Estado da arte

Com maior grau de evidência existem atualmente, outros dois *softwares* para controle de bandas de rede similares ao *Alternate Queueing - ALTQ*, são eles o *Dummy-*

[5] Segundo (CORREIA; SILVA, 2002), *bandwidth* ou banda passante, ou largura de banda ou *throughtput* é dada em função do número de *bits* que podem ser transmitidos sobre a rede em um certo período de tempo.

[6] *FreeBSD* - é um sistema operacional desenvolvido pela Universidade de Berkeley, California, que tem por ascendente o sistema operacional (*Unix*) *BSD-Berkeley Software Distribution*.

net que executa em ambiente *BSD*[7] e o *Traffic Control* - TC que executa em ambiente GNU-Linux. Existem também sistemas operacionais originados nos *BSDs* que possuem objetivos específicos - roteamento[8] e *firewall*[9] e trazem incorporados funções de controle de bandas. Incluem-se nesse caso as soluções *M0n0wall* e o *Pfsense*. Em contraponto acrescente-se possibilidades de utilização de soluções proprietárias para a função de controlar bandas, dentre estas incluem-se tradicionais fornecedores de equipamentos como *Cisco System* e *3Com*.

1.5.1 O *Dummynet*

O *Dummynet* é um programa para limitação de bandas de rede para o *FreeBSD*. O *Dummynet* tem como característica a intervenção em dois aspectos: a limitação de banda e controle de atrasos (*delay*). *Dummynet* tem algumas vantagens sobre o *Alternate Queueing - ALTQ*:

- - É implementado na camada *IP* do protocolo *TCP/IP*[10] - e não necessita de alterações de drivers;
- - Pode intervir no tráfego de entrada e de saída;
- - É implementado em conjunto com o *IPFW*[11].

Dentre as desvantagens em relação ao *Alternate Queueing - ALTQ* destacam-se:

- - O *Dummynet* não permite a implementação de disciplinas de filas;
- - O *Dummynet* é dependente e limitado ao tempo de resposta do *Kernel*[12].

Em resumo o *Dummynet* é útil para limitar bandas em ambientes não complexos e apresenta facilidades em sua configuração.

---

[7] Sistemas operacionais descendentes do *Unix-BSD*, tais como *FreeBSD*, *NetBSD*, *OpenBSD* dentre outros.

[8] Roteamento é a ação de equipamentos que interligam redes distintas

[9] Conforme (UCHOA, 2005)Firewall - é uma ferramenta de *software* ou *hardware* situada entre duas redes (uma interna e outra externa), responsável por filtrar pacotes, evitando o acesso externo a determinados serviços

[10] *TCP/IP -Transmition Control Protocol/Internet Protocol* protocolo de rede baseado em quatro camadas, desenvolvido pelo *DOD - Departament of Defense* inicialmente para fins militares, hoje amplamente utilizado

[11] *IPFW* é um filtro de pacotes nativo no kernel de sistemas operacionais *BSDs*.

[12] *Kernel* - núcleo do sistema operacional

1.5.2 O *Traffic Control - TC*

O *Traffic Control - TC* é bem semelhante ao *Alternate Queueing - ALTQ* pois implementa disciplinas de filas e com estas define suas operações.*Traffic Control - TC* é usado para moldagem (*shapping*[13]) do tráfego que entra com objetivo de melhorar o comportamento da rede, permite também o uso de disciplinas de filas para estabelecer prioridades.*Traffic Control - TC* é um ítem que demanda dificuldade pois, embora seja uma ferramenta *GNU-Linux*, não existem informações suficientes que facilitem sua implementação.

1.5.3 O *M0n0wall*

O *M0n0wall*[14] é um *software* com características de *firewall* e roteador baseado em um sistema operacional *FreeBSD* mínimo, pode ser utilizado em *PCs - Personal Computer* antigos, o que o torna uma solução pouco dispendiosa quando comparado a produtos comerciais. Sua administração é simples baseada em arquivos textos *XML-Extended Markup Language*. A principal potencialidade do *m0n0wall* é o controle de segurança, pois possue funções de *firewall* incorporadas e agrega funções de controle de banda através do *software Dummynet*, visto anteriormente, em conjunto com o *firewall - IPFW*.

1.5.4 O *Pfsense*

O *pfsense*[15] tal como o *m0n0wall* é baseado em um sistema operacional *FreeBSD* mínimo, sua finalidade também é segurança de acessos, porém é composto de um conjunto de *softwares* que o torna robusto para controle de bandas. Dentre estes softwares, inclue-se o *Alternate Queueing - ALTQ*, o *Packet Filter - PF*. A configuração do *pfsense* é simples e executada através de interfaces gráficas.

1.5.5 Soluções proprietárias

As soluções proprietárias são oferecidas por empresas tradicionais especialistas na fabricação de equipamentos e conjugados a estes *softwares* para integração de redes. A empresa *Cisco Systems* dota parte dos seus *roteadores* e *switches* do *software Cisco IOS - Input Output Subsystem* que permite realizar controle de banda. Estes equipamentos utilizam-se de disciplinas de filas tais como *WFQ* -

[13]Segundo (BROWN, 2003) - Shapping - é um mecanismo que retarda a entrega de pacotes com a finalidade de obter-se uma taxa de entrega desejável.É um mecanismo que utiliza-se de filas e é utilizado em soluções de controle de bandas de rede

[14]Detalhes em http://www.m0m0.ch/wall

[15]Detalhes em http://www.pfsense.org

Weighted Fair Queueing, CBQ - Class Based Queueing, da mesma maneira que os *softwares Traffic Control - TC* e o *Alternate Queueing - ALTQ*.

Dentre os equipamentos da empresa *Cisco Systems* que utilizam-se do *software Cisco IOS - Input Output Subsystem* cita-se:

- Roteadores - *Cisco 8xx*[16] *16xx, 17xx, 25xx, 36xx, 4xxx, 72xx, 75xx, 85xx e 12xxx*
- Switches - *Catalyst 4xxx, 5xxx e 6xxx*

No presente trabalho será utilizado o *Alternate Queueing - ALTQ* por conveniências meramente profissionais. O ambiente de trabalho onde será executada a implementação possue máquinas executando o sistema operacional *FreeBSD*. Utiliza-se também o *Packet Filter - PF* como *firewall* e não apresenta nenhum mecanismo de intervenção para controle de tráfego. E pretende-se testar e usar uma ferramenta de controle de tráfego que esteja totalmente integrada ao ambiente operacional adotado.

1.6 Organização do trabalho

No capítulo 2 descreve-se o *software Alternate Queueing - ALTQ*, o histórico de seu desenvolvimento, o desenho e seus objetivos, a integração com o *Packet Filter - PF* e introduz-se o capítulo 3 - As filas.

O capítulo 3 é dedicado as filas, elementos da arquitetura das filas, disciplina de filas e algoritmos de controle de congestionamentos, tendo em vista serem pontos centrais nos *softwares* que tem por objetivo o controle de bandas.

O capítulo 4 é dedicado a parâmetros de configuração do *Alternate Queueing - ALTQ* descrevendo-se como usá-los.

O capítulo 5 discute-se um ambiente operacional para implementação do *Alternate Queueing - ALTQ*, a construção do arquivo de regras para o *software* nesse ambiente, os parâmetros que serão usados para avaliação da efetividade do *software* e a forma com que os dados serão obtidos .

O capítulo 6 com os dados já obtidos e as regras estabelecidas, procede-se a análise dos dados demosntrando-se a efetividade do *software* no ambiente operacional.

O capítulo 7 é a conclusão do trabalho e a possibilidade de trabalhos futuros sobre o tema tratado.

[16] O equipamento *Roteador Cisco 805* destinado a empresas médias e escritórios é vendido por R$ 2.800,00 - Loja Cisco (www.lojacisco.com.br) 16-08-2006

Capítulo 2

O *Alternate Queueing - ALTQ*

2.1 Histórico

O *Alternate Queueing - ALTQ* é um *software* desenvolvido por Kenjiro Cho pesquisador do Sony Computer Science Laboratories Inc.[1]. O *software* está vinculado ao projeto Kame[2] que teve início em abril de 1998. O projeto Kame foi uma iniciativa do governo japonês e tinha por finalidade juntar esforços para produção de ferramentas simples e robustas tendo em vista os protocolos *IPv6*[3] e *IPsec*[4] e implementações avançadas para redes tais como enfileiramento de pacotes, mobilidade e outras.

2.2 Objetivos

O segundo (CHO, 2001) o objetivo do *Alternate Queueing - ALTQ* é prover uma plataforma flexível que possa suportar os diversos tipos de funções de *QoS - Quality of Service*[5] de uma rede. Deste objetivo central derivou a possibilidade de gerenciamento de bandas de redes através de controle de tráfego de saída.

---

[1] Vide `http://www.csl.sony.co.jp/`

[2] Vide `http://www.kane.net/`

[3] Segundo (WIKIPEDIA, 2006), *IPv6* é a versão 6 do protocolo *IP* que tem como objetivo substituir o padrão anterior, o *IPv4*, que só suporta cerca de 4 bilhões (4 x 109) de endereços, enquanto que o IPv6 suporta 3.4 x 1038 endereços.

[4] *IPsec - software* agregado ao protocolo *IP*com finalidades de segurança

[5] Segundo (FARIA; SANTOS, 2005)*QoS - Quality of Service* - qualidade de serviços em redes de computadores consiste na implementação de mecanismos ou modelos que permitam diferenciação de tráfego

2.3 Desenho

A solução usada pelo *Alternate Queueing - ALTQ* para gerenciar a largura de banda é baseada na mudança do mecanismo FIFO[6] implementado nos sistemas operacionais, que processam os pacotes na ordem em que eles chegam. Isto é feito de forma simples: Com *Alternate Queueing - ALTQ* os pacotes são designados para filas com diferentes prioridades. Pacotes em filas com maior prioridade são processados antes daqueles com menor prioridade, que são mantidos em memória até que não exista pacotes de maior prioridade. O *Alternate Queueing - ALTQ* foi desenhado de maneira a fazer mínimas mudanças no *kernel*. Este porém, não possue abstrações suficientes para implementar os vários tipos de disciplinas de filas[7]. Assim existem muitas partes no *kernel* que operam com a disciplina *FIFO*.

As disciplinas de filas portadas para o *Alternate Queueing - ALTQ* são:

- PQ -*"Priority Queueing"*
- CBQ -*"Class Based Queueing"*
- WFQ -*"Weighted Fair Queueing"*
- SFQ -*"Stochastic Fairness Queueing"*
- HFSC -*"Hierarquical Fair Service Curve"*

Em detalhe na Figura 2.1 verifica-se a alteração promovida pelo *Alternate Queueing - ALTQ* para implementar as disciplinas de filas.

2.4 Integração do *Alternate Queueing - ALTQ* com o *Packet Filter - PF*

A partir das versões *5.x/6.x do FreeBSD* o *Alternate Queueing - ALTQ* foi integrado ao *Packet Filter - PF*, desta forma as regras de configuração do *Alternate Queueing - ALTQ* serão escritas no arquivo *pf.conf* (localizado normalmente no diretório /etc) juntamente com as regras do *Packet Filter - PF*. Tal fato, facilitou o trabalho do Administrador de rede permitindo-lhe ajustar o controle de tráfego com as regras de filtragem do *firewall*.

[6] FIFO:*First in first out* - O primeiro a entrar é o primeiro a ser processado
[7] Vide: 3.3 - Disciplina de filas

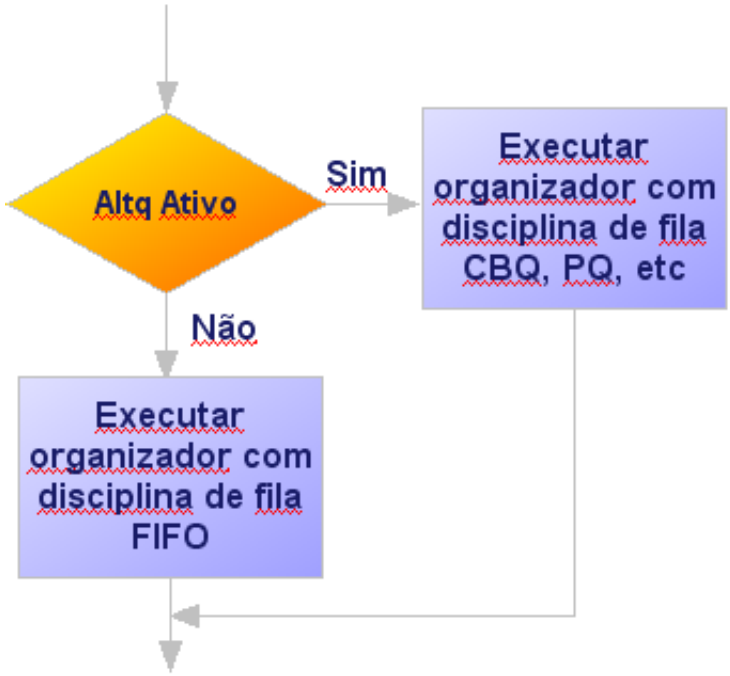

Figura 2.1: Detalhe da alteração do *Alternate Queueing - ALTQ* no Kernel

2.5 O *Alternate Queueing - ALTQ* e as disciplinas de filas

O modo de operação do *Alternate Queueing - ALTQ* consiste em desviar o fluxo de saída de informações da disciplina de filas *FIFO - First in First out* para outras disciplinas de filas que permitam melhores controles. Dessa forma o conhecimento prévio sobre as filas e as disciplinas de filas é importante para que se possa configurar o *software* adeqüadamente.

Capítulo 3

As Filas

3.1 Fundamentos de filas

A teoria das filas é o estudo análitico do comportamento das filas e é um conceito de auxílio no desenvolvimento nos mais variados sistemas (sistemas de atendimentos, sistemas viários), e também de desempenho em sistemas de computadores. A teoria das filas é antiga, suas as aplicações práticas voltaram à tona com o aparecimento dos computadores digitais.

Implementações de controle de bandas ou controle de tráfego são baseados na teoria das filas. Filas são locais onde se armazenam algo respeitando-se uma determinada ordem para um processamento posterior. A comunicação em uma rede de computadores é feita através de troca de pacotes de dados entre eles. Esses pacotes aguardam o processamento em filas. A ordem de processamento desses pacotes pode afetar o desempenho da rede de computadores. As aplicações em que o "tempo de resposta" é fator crítico, devem ter prioridade de processamento maior que as demais. O ordenamento e as prioridades são aspectos importantes para o processamento das filas.

3.2 Componentes de Filas

Filas é um termo genérico para designar-se uma arquitetura de filas e representam um dos últimos componentes dessa arquitetura. Uma arquitetura de filas é representada por blocos funcionais ou componentes, onde o fluxo passa, que executam determinado serviço, tal como identificação, medição e outros. A Figura 3.1 mostra uma visão geral de uma arquitetura de filas e seus componentes.

São componentes de uma arquitetura de filas:

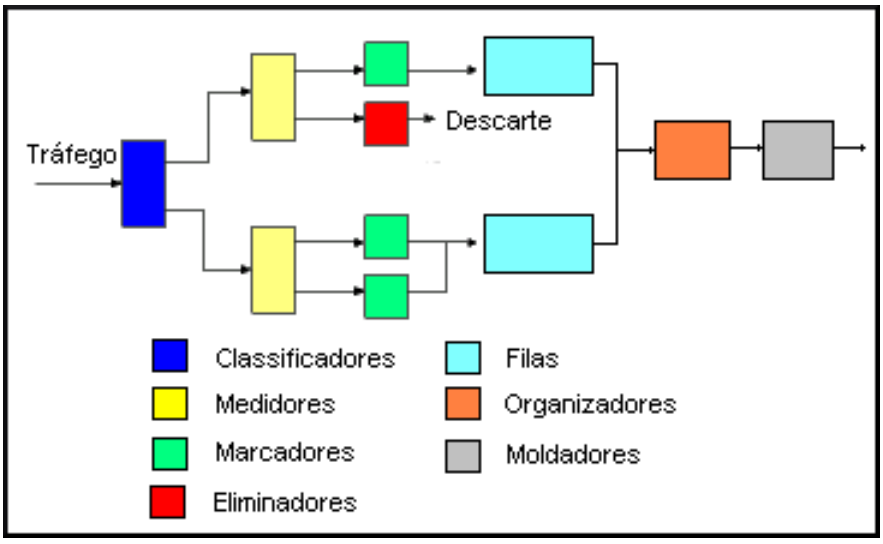

Figura 3.1: Visão geral de uma arquitetura de filas

Classificadores - identificam pacotes baseando-se em algum argumento como o endereço origem, o endereço destino, o número de porta, o protocolo e outros. Os pacotes identificados são classificados para processamento posterior.

Medidores - tem a finalidade de medir o fluxo de tráfego de pacotes. Os resultados da medição serão usados por outros elementos.

Marcadores - gravam o pacote com um valor determinado. Esse valor pode ser uma prioridade, uma informação de congestionamento, algum tipo de aplicação e outros.

Eliminadores - descartam alguns ou todos os pacotes do fluxo de maneira a limitar quantidade de pacotes nas filas, ou eliminar congestionamentos .

Filas - são *buffers* finitos para armazenamento de pacotes para entrega futura.

Organizadores - verificam qual fila processar e como deve ser o processamento da fila. Os sistemas operacionais usam por padrão um organizador processando na disciplina de filas *FIFO*. Porém existem outras disciplinas que permitem um maior controle de banda ou controle de tráfego.

Moldadores - atrasam alguns ou todos os pacotes no fluxo de forma a limitar a taxa de pico do fluxo. Tem um *buffer* finito e os pacotes podem ser descartados caso não haja espaço nesses *buffers*.

3.3 Disciplinas de filas

Tal como o *schedulle* no tópico "Escalonamento de Processos"em (TANENBAUM, 1992), uma disciplina de filas é a faculdade de manipular as filas de maneira a conciliar exigências contraditórias tais como:

- Imparcialidade - garantia que toda a informação tenha chances iguais de serem processadas;
- Proteção - garantia do não descarte da informação;
- Limites de desempenho - garantia de plena utilização dos meios, sem que haja degradação;
- Facilidade de implementação ou administração - simplicidade de configuração e administração.

O principal objetivo de uma disciplina de filas é a distribuição da largura de banda, permitindo a possibilidades de priorizações e isolamento de tráfegos mal comportados.

A implementação de uma disciplina de filas é avaliada por simulação para depois ser portada para os *softwares* de controle de tráfego. Para implementar-se uma nova disciplina de filas deve-se atentar-se para as rotinas de enfileiramentos (*enqueue*) e *desfileiramentos* (*dequeue*) e toma-se sempre por base de desenvolvimento (*template*) a implementação *FIFO*. Concluído o desenvolvimento, a nova disciplina pode ser integrada ao *software* de controle de tráfego, neste estudo o *Alternate Queueing - ALTQ*, através da inclusão na *ALTQ device table*.

A seguir, as disciplinas de filas portadas para o *Alternate Queueing - ALTQ*:

3.3.1 *FIFO - First in First out*

A disciplina de filas *FIFO -First in, First out* - primeiro que entra, primeiro que sai tem como característica a simplicidade pois somente uma fila é processada e os pacotes são processados obedecendo-se a ordem de chegada. Não provê decisões sobre prioridades de tráfego, a ordem de chegada determina a largura da banda. Caso haja aumento de tráfego de rede acima da capacidade da fila, haverá necessidade de descarte de informações. Se a situação tornar-se freqüente a

degeneração dos serviços de rede é patente. As redes atuais operam com disciplinas de filas mais sofisticadas. Como mencionado anteriormente a disciplina de filas *FIFO -First in, First out* é usada como modelo (*template*) para construção de novas disciplinas de filas.

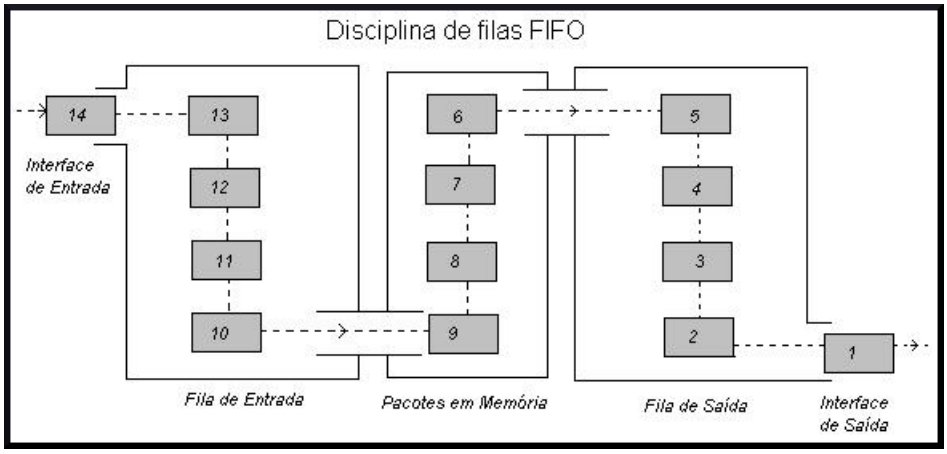

Figura 3.2: Disciplina de filas *FIFO -First in First out*

3.3.2 Filas baseadas em classes - *CBQ - Class Based Queueing*

CBQ - Class Based Queueing é a disciplina de fila que caracteriza-se em dividir a porção de banda rede em filas ou classes. O tráfego que se atribui a estas classes é obtido observando-se campos do pacote de dados tais como endereço de origem ou destino, número de porta, protocolo e outros. Existe uma hierarquia dentro das filas (classes) obtidas em função da prioridade de cada uma, e pode haver filas que se beneficiam de empréstimos de banda de outras filas quando houver necessidade. Esta hierarquia pode ser extendida nos níveis menores.

Com a disciplina *CBQ - Class Based Queueing* é possível garantir uma largura de banda em um ponto potencial de congestionamento através da especificação de uma largura fixa, deixando a outra porção da banda para outros tráfegos. É considerado um modelo eqüitativo, com o aumento de recursos para as filas de mais alta prioridade e uma diminuição relativa para as filas de mais baixa prioridade. *CBQ - Class Based Queueing* é considerada uma disciplina eficiente para gerenciamento de recursos de filas. Existem problemas como a sobrecarga computacional gerada pela reordenação dos pacotes.

Segue na Tabela 3.1 abaixo uma estrutura de classes hipotética baseada na disciplina *CBQ - Class Based Queueing*.

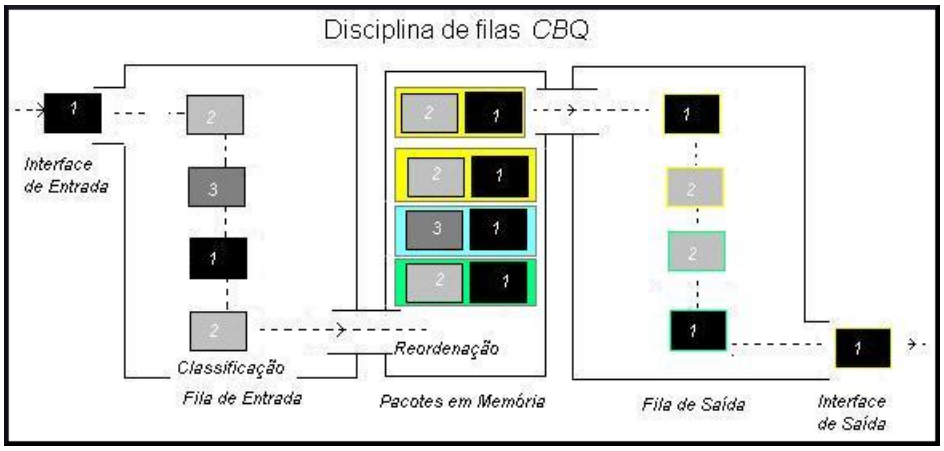

Figura 3.3: Disciplina de filas *CBQ - Class Based Queueing*

Tabela 3.1: Estrutura de filas/classes na disciplina *CBQ - Class Based Queueing*

Fila Pai(3 Mbps)			
	Fila A (2 Mbps)		
		Fila AA (200 Kbps)	
		Fila AB (1800 Kbps, borrow)	
	Fila B (1 Mbps)		
		Fila BA (250 Kbps)	
		Fila BB (750 Kbps)	
			Fila BBA (100 Kbps)
			Fila BBB (650 Kbps)

O parâmetro *borrow* indica a possibilidade da *fila AB* ser beneficiada por empréstimos de banda da *fila AA* caso esta não esteja usando toda a quantidade que lhe foi determinada (*200 Kbps*).

Uma das características mais importantes da disciplina de filas *CBQ - Class Based Queueing* é a de poder-se atribuir prioridades à classes ou filas, onde as classes ou filas com prioridades maior são as mais beneficiadas nos casos em que haja congestionamento. As filas com mesma prioridade serão processadas em seqüência. Segue-se na Tabela 3.2 abaixo uma estrutura de classes com prioridades onde usa-se o parâmetro *priority* para a atribuição de prioridades.

As filas *Fila* A e *Fila B* serão processadas em seqüência, porém no processamento da *fila A* haverá processamento com prioridades de *subfilas*, onde se processará inicialmente a *fila AA* e posteriormente a *fila AB*.

Tabela 3.2: Estrutura de filas/classes na disciplina *CBQ - Class Based Queueing* - prioridades

Fila Pai(5 Mbps)		
	Fila A (3 Mbps, priority 1)	
		Fila AA (1200 Kbps, priority 3)
		Fila AB (1800 Kbps, priority 2)
	Fila B (2 Mbps, priority 1)	
		Fila BA (250 Kbps)
		Fila BB (1750 Kbps)

3.3.3 Filas com prioridades - *PQ - Priority Queueing*

A disciplina *PQ - Priority Queueing* caracteriza-se por permitir mútiplas filas associadas com diferentes prioridades, onde a fila de maior prioridade é processada primeiro. Esta é uma disciplina mais simples que a *CBQ - Class Based Queueing* por não permitir o uso de *subfilas*. Dentre as vantagens desta disciplina destaca-se a possibilidade de atribuir-se prioridades para as famílias de protocolos, priorizandos-se o tráfego *TCP* em detrimento de outros (por exemplo). Porém se o volume de tráfego que tiver maior prioridade for alto, os demais tráfegos poderão ser prejudicados. A reordenação dos pacotes de saída pelo realinhamento de prioridades pode degradar o processador principalmente quando existir aumento de velocidade no canal.

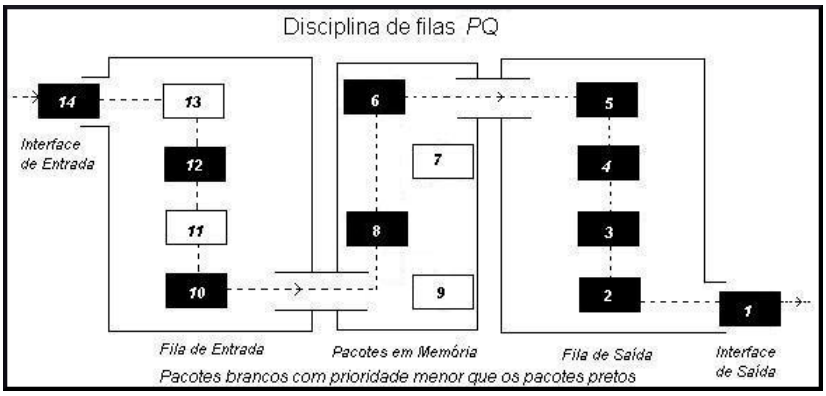

Figura 3.4: Disciplina de filas *PQ - Priority Queueing*

Na Tabela 3.3 verifica-se uma estrutura da disciplina *PQ - Priority Queueing*. A *Fila D* é processada primeiro, quando toda esta fila for processada ou esteja vazia a *fila C* será processada. Deve-se planejar cuidadosamente as filas pois existe a possibilidade daquelas com prioridade menor tardarem a serem processadas e as-

Tabela 3.3: Estrutura de filas/classes na disciplina *PQ - Priority Queueing*

Fila Pai(5 Mbps)	
	Fila A (2 Mbps, priority 1)
	Fila B (1 Mbps, priority 2)
	Fila C (1 Mbps, priority 3)
	Fila D (1 Mbps, priority 4)

sim haja descarte de pacotes trazendo impactos de desempenho em outros serviços de rede. Os pacotes nas filas desta disciplina são processadas através da disciplina *FIFO - First in First out*.

3.3.4 Filas Justas *WFQ - Weighted Fair Queueing*

WFQ - Weighted Fair Queueing - é uma disciplina de fila que admite uma fila independente para cada fluxo. *WFQ - Weighted Fair Queueing* - proporciona alocação de banda imparcial em congestionamentos e protege os fluxos já que existe distribuição proporcional à capacidade de rede entre eles. *WFQ - Weighted Fair Queueing* - apresenta comportamento previsível na entrega de pacotes dando aos fluxos de baixo volume de tráfego tratamento preferencial. Assim cargas de menor volume são transmitidas em uma porção de tempo menor e os fluxos de maiores volumes usam o restante da capacidade de enfileiramento. *WFQ - Weighted Fair Queueing* - foi projetada para que haja o mínimo esforço na configuração, adaptando-se automaticamente às mudanças das condições de tráfego de rede.

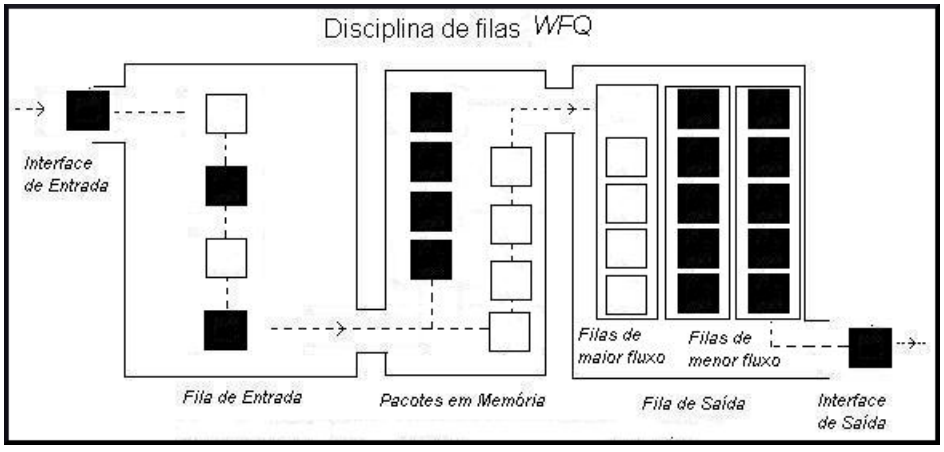

Figura 3.5: Disciplina de filas *WFQ - Weighted Fair Queueing*

3.3.5 Outras disciplinas de filas

SFQ - Stochastic Fairness Queueing

A disciplina *SFQ - Stochastic Fairness Queueing* é similar a *WFQ - Weighted Fair Queueing*, porém utiliza-se de uma função "hash" para mapear e distribuir banda entre os fluxos.

HFSC - Hierarquical Fair Service Curve

A disciplina *HFSC - Hierarquical Fair Service Curve* também deriva da disciplina *WFQ - Weighted Fair Queueing*, a diferença principal entre as duas disciplinas é a melhor capacidade de gerenciamento proporcionada por *HFSC*.

3.4 Algoritmos de controles de congestionamentos

Além das disciplinas de filas, existem outros mecanismos conhecidos como detectores explícitos de congestionamentos, segundo (ANDRADE *et al.*, 2003) os congestionamentos devem ser notificados de maneira rápida ao transmissor de modo a não sobrecarregar uma rede destino. O transmissor não têm como detectar a capacidade de transferência do destino de modo a que possam transmitir à uma taxa compatível. Utiliza-se a perda de pacotes como um indicador de congestionamento, esta forma de indicação é caracterizada como implícita, pois o congestionamento é inferido.

Dentre as iniciativas de controle de congestionamentos cita-se o *ECN - Explicit Congestion Notification* e o *RED - Randon Early Detction*.

3.4.1 ECN - Explicit Congestion Notification

O *ECN - Explicit Congestion Notification* descrita em (RAMAKRISHMAN; FLOYD; BLACK, 2001) - é um dispositivo indicador de congestionamentos. *ECN - Explicit Congestion Notification* notifica o emissor a existência de congestionamento através de *bits* no cabeçalho[1] do pacote. *ECN - Explicit Congestion Notification* é efetivo quando o congestionamento é mínimo pois reduz retransmissões reduzindo o tráfego de rede.

[1] Ainda conforme (ANDRADE *et al.*, 2003), o *ECN* utiliza-se de dois *bits* do cabeçalho *IP* e dois do cabeçalho *TCP*. No cabeçalho *IP* são os *bits* 6 e 7 (antigo campo *TOS - Type of Service*, agora *ECT - (ECN Capable Transport)* e do campo *CE - Congestion Experienced*. No cabeçalho *TCP*, os *bits* 8 e 9 do campo reservado são usados como *CWR - Congestion Window Reduced* e *EC - ECN-Echo*

3.4.2 *RED - Randon Early Detection*

O *RED - Randon Early Detection* (FLOYD; JACOBSON, 1993) é um algoritmo que tem por finalidade evitar congestionamentos através de observações nas filas. *RED - Randon Early Detection* examina um valor mínimo e um valor máximo pré-determinados e faz comparações com um valor médio calculado em função do tráfego. Se o valor médio estiver abaixo do mínimo nenhum pacote será descartado, caso contrário todos os pacotes que chegarem serão descartados. O *RED - Randon Early Detection* quando detecta congestionamentos utiliza-se do *ECN - Explicit Congestion Notification* para sinalizá-lo.

Capítulo 4

O *Configuração do Alternate Queueing - ALTQ*

4.1 O *Alternate Queueing - ALTQ* no *FreeBSD*

Para que o *Alternate Queueing - ALTQ* execute no sistema operacional *FreeBSD*, o seu *kernel* deve estar compilado com as seguintes opções:

options ALTQ

options ALTQ_CBQ # Class Bases Queueing

options ALTQ_RED # Random Early Detection

options ALTQ_RIO # RED In/Out

options ALTQ_HFSC # Hierarchical Packet Scheduler

options ALTQ_CDNR # Traffic conditioner

options ALTQ_PRIQ # Priority Queueing

#options ALTQ_NOPCC # Required for SMP build

options ALTQ_DEBUG

4.2 Parâmetros de operação do *Alternate Queueing - ALTQ*

O *Alternate Queueing - ALTQ* foi concebido para executar em sistemas operacionais *BSD*, e necessitava anteriormente de um arquivo de configuração para

determinar-se as filas e suas prioridades. Com a integração do *Alternate Queueing - ALTQ* com o *Packet Filter - PF* as configurações dos *softwares* passaram a ser feitas em conjunto o que facilitou sua operação. Assim os parâmetros usados para a configuração do *Alternate Queueing - ALTQ* bem como do *firewall Packet Filter - PF*, devem ser declarados no arquivo "pf.conf" que encontra-se no diretório "/etc". Dessa forma, para se habilitar e operar o *Alternate Queueing - ALTQ* é necessário utilizar as diretivas *altq* e *queue* cuja sintaxes são descritas abaixo:

4.2.1 Diretiva *altq*

A diretiva *altq* habilita o *Alternate Queueing - ALTQ* em uma determinada *interface*[1] de rede, definindo a disciplina de filas, as filas e a quantidade total da banda.

altq on *interface disciplina bw qlim tamanho filas*

Segue-se a descrição dos parâmetros:

altq on - Obrigatório para ativar o *Alternate Queueing - ALTQ*;

interface - Interface de rede onde haverá o controle;

disciplina - Disciplina de fila a ser utilizada, somente uma disciplina por vez;

bw - Valor total da banda de rede deve ser antecedida da palavra chave *bandwithd*;

qlim - Pacotes a serem armazenados na fila, deve ser precedida da palavra *qlimit*, valor *default* em 50;

tamanho - Opcional, baseado na largura de banda da interface, se usada deverá ser precedida pela palavra *tbrsize*;

filas - Lista de sub-filas a serem criadas sob a fila raiz, dever ser precedida pela palavra *queue*.

Assim: *altq on rlx0 cbq bandwith 2Mb queue {*filaA, filaB, filaC*}*

Habilitará a disciplina *Class Based Queueing - CBQ* na interface de rede *rlx0*, onde a largura total de banda é 2 Mbps e onde 3 filas são definidas *filaA*, *filaB* e *filaC*.

[1] Conforme (UCHOA; SICA; SIMEONE, 2003), devido a grande variedade de *hardware* disponível no mercado, o protocolo *TCP/IP* definiu uma interface (virtual), através da qual o sistema acessa o *hardware*.

4.2.2 Diretiva *queue*

A diretiva *queue* determina como vai operar as filas definidas na diretiva *altq*, qual vai ser a amplitude de cada fila, suas prioridades e subfilas

queue *nome on interface* bandwidth *bw* priority *pri* qlimit *qlim disciplina opções da disciplina* { lista de filas}.

Segue a descrição de parâmetros da diretiva *queue*.

nome - Nome da fila, que deve ser um dos nomes definidos na diretiva *altq*;

interface - Interface de rede onde ocorrerá a fila;

bw - A quantidade de largura de banda para a fila;

pri - A prioridade da fila, Para *CBQ - Class Based Queueing* pode ser de 0 até 7 e para *PRIQ* de 0 a 15;

qlim - O número máximo de pacotes que se pode armazenar na fila;

disciplina - A disciplina utilizada;

opções de disciplinas - Opções a serem passadas para o controle do comportamento da disciplina. Inclui-se nas opções os mecanismos de controle de congestionamentos explícitos como *RED - Randon Early Detection* e *ECN - Explicit Congestion Notification*. Dentre as opções registra-se também a possibilidade de empréstimos de bandas através do parâmetro *borrow* e a atribuição de fluxos não definidos em outras filas para a fila com opção *default*;

lista de filas - A lista de sub-filas que podem ser criadas nesta fila, válido somente com a disciplina *CBQ - Class Based Queueing*.

4.2.3 Atribuição de tráfego às filas

O *Packet Filter - PF*

A habilitação do *Alternate Queueing - ALTQ* em determinada *interface* de rede com uma determinada disciplina de fila através da diretiva *altq*, a atribuição de tráfego às filas definidas na diretiva queue é feita através de inclusão de regras para o *Packet Filter - PF*, um firewall para sistemas operacionais *FreeBSD*, *NetBSD* e *OpenBSD*. Conforme (OPENBSD, 2006a), o *Packet Filter - PF* caracteriza-se por ser um filtro de pacotes que bloqueia ou libera a passagem de pacotes nas interfaces de rede. Os critérios para sua ação são baseados nos *headers* da camada 3

(protocolos *IPv4* e *IPv6*) e da camada 4 (protocolos *TCP*, *UDP*, *ICMP* e *ICMPv6*). Os critérios que são mais comumente usados na interceptação dos pacotes são o endereço origem e o endereço destino, a porta origem e a porta destino e os protocolos.

Quando ocorrer interceptação de um pacote deve-se tomar uma ação: ou permitir, ou negar sua passagem. As regras são avaliadas em ordem seqüencial, da primeira para a última. Pode-se especificar uma maneira para que a avaliação seja interrompida em um determinada regra e não prossiga até o final. A última regra que satisfazer o critério é aquela que informará a ação a ser tomada.

Sintaxe das regras

A sintaxe de uma regra tem o seguinte modelo simplificado:

ação [direção] [log] [quick] [on interface] [fa] [protocolo] [endereço origem] [porta origem] [endereço destino] [porta destino] [alertas TCP] [estado] [fila]

ação - A ação a ser tomada quando os pacotes são interceptados ou *pass*, ou *block*. A ação *pass* passará o pacote para o *kernel* para o processamento seguinte, enquanto que a ação *block*, poderá excluir o pacote sem nenhum aviso (*block drop*) ou excluirá e enviará um aviso *block return*;

direção - A direção em que o pacote está se movimentando em relação a interface *in* para a interface, ou *out* saindo da interface;

log - Especifica se o pacote deve ser gravado, caso na opção estado estiver configurado *keep state*, *modulate state* ou *synproxy state*, somente o pacote que estabeleceu a coneção será gravado. Para gravar indistintamente todos o registros usa-se *log (all)*;

quick - Quando um pacote for interceptado pela regra com o parâmetro *quick*, não haverá mais avaliações de outras regras;

on interface - O nome ou grupo de uma interface no qual o pacote está se movimentando. Um grupo de interface pode ser especificada pelo seu prefixo, por exemplo *rl* ou *ppp*;

fa - A família do pacote *inet* para IPv4 ou *inet6* para *IPv6*. O *software* determina este parâmetro baseando-se no endereço fonte ou de destino;

protocolo - Um protocolo da camada 4 da pilha *TCP/IP* (*tcp*, *udp*, *icmp*, *icmp6*), um protocolo válido contido em */etc/protocols*, um número de protcolo entre 0 e 255, ou um grupo de protocolo definido em uma lista;

endereço origem - É o endereço origem contido no *header* do pacote IP;

porta origem - É a porta origem;

endereço destino - É o endereço destino contido no *header* do pacote IP;

porta destino - É a porta destino;

alertas TCP - Para uso em conjunto com *proto tcp* identifica *flags* do protocolo, tais como *SIN*, *ACK*;

estado - Especifica se o estado dos pacotes devem ser mantidos com pacotes que coincidam com a regra;

fila - Especifica qual a fila é para ser destinado o tráfego capturado.

Na seção Anexos A.3 - “Aplicações das Diretrizes” construiu-se um ambiente hipotético com uso das diretrizes e seus parâmetros.

Capítulo 5

Materiais e Métodos

5.1 Implementação de controle de banda com *Alternate Queueing - ALTQ*

É importante ter-se instrumentos que permitam o controle do uso de banda de redes através de intervenções do administrador. Para avaliar-se a efetividade do *software Alternate Queueing - ALTQ* como um destes instrumentos, definiu-se um ambiente operacional cuja finalidade foi a geração de tráfego intermediado pelo *Alternate Queueing - ALTQ*. Posteriormente, promoveu-se alterações em parâmetros de controle, observou-se os dados gerados, e finalmente realizou-se a análise destes dados.

5.2 Ambiente Operacional

O ambiente operacional (Figura 5.1) para o experimento é formado por dois computadores interconectados através de um *hub*[1].O computador denominado *onix* executa o sistema operacional FreeBSD versão 5.4 com os *softwares Alternate Queueing - ALTQ, Packet Filter - PF*, e o *Netperf 2.4.1*. O segundo computador denominado *diamante* executa o sistema operacional GNU-Linux Conectiva 10, com o *software Netperf 2.4.1*.

5.3 Arquivo de regras

O arquivo de regras (pf.conf) que é comum ao *Packet Filter - PF* e ao *Alternate Queueing - ALTQ* está relacionado em A.1 */etc/pf.conf*. No arquivo de regras

[1] *Hub* - é um equipamento que tem por finalidade interligar computadores de uma rede local

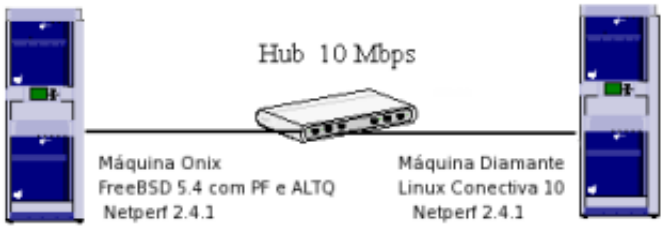

Figura 5.1: Ambiente operacional

definiu-se as filas, a disciplina de filas a ser usada no experimento, e a porcentagem inicial de uso de cada fila (parâmetro *bandwidth*) já que pretende-se alterar estes percentuais de uso a cada ciclo do experimento. Optou-se por usar a disciplina de filas *CBQ - Class Based Queue* pela facilidade de implementação dos parâmetros e pelos recursos adicionais que esta disciplina oferece (embora estes não sejam utilizados no experimento).

5.4 Parâmetros e regras para avaliação de efetividade

Para a avaliação de efetividade do *Alternate Queueing - ALTQ* no controle de bandas de rede espera-se que:

- As alterações de parâmetros promovidas a cada ciclo do experimento para aumento ou diminuição da largura de banda de determinada fila (ou classe) tenham o correspondente aumento ou diminuição da vazão *througput* para aquela fila (ou classe).

5.5 Forma de obtenção dos dados

5.5.1 Obtenção dos dados de filas

Para a obtenção de dados definiu-se duas filas no arquivo de configuração (A.1 */etc/pf.conf*) do *Packet Filter - PF*:

- a fila *q_tcp* - para o tráfego *TCP* de saída;
- a fila *q_ssh* - para o tráfego *SSH* de saída;

A geração de dados para o tráfego *TCP*, será efetuado através do *software netperf* que é uma ferramenta de *benchmarking* que mede desempenho de redes. O *netperf*

versão 2.4.1 é composto por dois módulos: o *netperf* e o *netserver*. O tráfego gerado pelo módulo *nerperf* da máquina *onix* é transmitido para a máquina *diamante* que está executando o *netserver* que retorna o tráfego para o gerador. A terminologia *netperf* é a definida em (BRADNER, 1991). Assim, observando-se a Figura 5.1, na estação *refletora* (*diamante*) executa-se o *netserver* e na estação *geradora* (*onix*) do tráfego executa-se o *netperf*.

Detalhes da execução do *netperf* em *onix*:

```
netperf -l sss -t TCP_RR -H 168.0.0.1 - -r xxx,yyy
```

onde

-l : tempo de duração da geração do tráfego
-t : protocolo e tipo de medição, no caso Request/Response do TCP
-H : endereço ip da máquina remota que executa o netserver
-r : tamanho do segmento de dados de envio e de resposta

5.5.2 Obtenção da vazão entre as máquinas *onix* e a *diamante*

Executa-se *netserver* em *diamante* (computador refletor) e executa-se o script A.2.1 na máquina *onix*, obteve-se vinte resultados e apurou-se a média da vazão entre as duas máquinas (em torno de 7,4 Mbps). Este valor será utilizado na diretiva *ALTQ* do arquivo de regras.

Assim a diretiva *altq* do pf.conf deverá ser:

```
altq on $ext_if bandwidth 7.4Mb cbq queue { q_tcp, q_ssh }
```

5.5.3 Obtenção de dados da fila q_tcp

Para a obtenção de dados na fila *q_tcp*, gera-se tráfego sem que o *Alternate Queueing - ALTQ* esteja ativo. Posteriormente ativa-se o *Alternate Queueing - ALTQ* e gera-se dados *TCP* e *SSH* separadamente. Na geração de tráfego *TCP* deixa-se a fila *q_tcp* com *bandwidth* 5% e a fila *q_ssh* com *bandwidth* 95%. Ao final de cada ciclo (execução do script A.2.2), incrementa-se 5% no parâmetro *bandwidth* da fila do tráfego *TCP* e decrementa-se 5% na fila de tráfego *SSH*, até o *bandwidth* do *TCP* atingir 100% e a do *SSH* se tornar nulo.
Inicialmente a definição da fila q_tcp será:

```
queue q_tcp bandwidth 5% priority 2 cbq
```

E a fila *q_ssh* será definida no *pf.conf*:

```
queue q_ssh bandwidth 95% priority 1 cbq(default)
```

O processo está descrito no fluxograma (Figura 5.2):

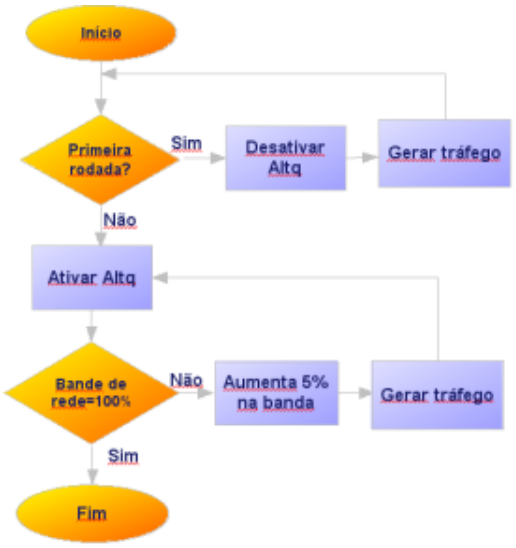

Figura 5.2: Forma de obtenção dos dados

A execução do script A.2.2 permite gerar dez resultados para cada *bandwitdh* (de 5% até 100%). Na análise dos dados será usado a média geométrica desses dez resultados por *bandwidth*. Assim obtém-se 20 resultados de vazão *throughput* para a fila *q_tcp*.

5.5.4 Obtenção de dados da fila q_ssh

Gera-se dados *SSH* para a fila *q_ssh*, através da transferência de um arquivo de 98Mb por *sftp*[2] da máquina *onix* para a máquina *diamante*. Para cada transferência efetua-se o incremento/decremento de 5% nas filas *q_ssh* e *q_tcp* respectivamente. O incremento passa ser feito na fila q_ssh a partir de 5% até atingir 100%, o decremento é efetuado na fila q_tcp.

Inicialmente a definição da fila *q_ssh* será:

[2] *sftp* - utilitário semelhante ao *ftp* usado para transferências de arquivos

```
queue q_ssh bandwidth 5% priority 2 cbq
```

E a fila *q_tcp* será definida no *pf.conf*:

```
queue q_tcp bandwidth 95% priority 1 cbq(default)
```

5.5.5 Interceptação das filas *TCP* e *SSH*

A interceptação dos tráfegos *TCP* e *SSH* para as filas *q_tcp* e *q_ssh* é descrita no arquivo *pf.conf*, da seguinte maneira:

Interceptação do tráfego *TCP* para a fila *q_tcp*:

```
pass in on $ext_if proto tcp from any to $ext_if keep state queue
(q_tcp)
```

```
pass out on $ext_if proto tcp from any to $ext_if keep state queue
(q_tcp)
```

Interceptação do tráfego *SSH* para a fila *q_ssh*:

```
pass in on $ext_if proto tcp to any port 22 keep state queue (q_ssh)
```

```
pass out on $ext_if proto tcp to any port 22 keep state queue (q_ssh)
```

5.6 Limitações do experimento

No experimento utiliza-se dois computadores interconectados por um *hub* operando à 10Mbps, pertencente a uma rede local e sofrendo influência de seu tráfego (outras vinte e duas máquinas). O ambiente ideal é ter-se máquinas segregadas e utilizar-se *hubs* inteligentes (*switches*), e assim obter-se medições sem influências externas.

Capítulo 6

Resultados obtidos

6.1 Dados obtidos

6.1.1 Dados da vazão (*throughput*) *onix-diamante*

Através da execução do *script* A.2.1 obtem-se os dados constantes na Tabela 6.1 referente a vazão (*througput* entre as máquinas *onix* e *diamante*. Gerou-se vinte medições e obteve-se a vazão média que é de 7,4 Mbps.

6.1.2 Dados obtidos pela geração de tráfego *SSH*

Na Tabela 6.2 verifica-se os resultados de gerações de tráfegos *SSH* obtidos através de transferência de arquivo de 96M (por *sftp*) da máquina *onix* para a máquina *diamante* com acréscimos de 5% na fila *q_ssh*, e decréscimos de 5% na fila *q_tcp*.

6.1.3 Dados obtidos pela geração de tráfego *TCP* através do software *Netperf*

Na Tabela 6.3 observa-se as médias dos resultados de gerações de tráfegos *TCP* através do software *Netperf* executando-se o script A.2.2 com acréscimos de 5% na fila *q_tcp* e decréscimos de 5% na fila *q_ssh* a cada ciclo.

6.2 Análise dos dados obtidos

6.2.1 Tráfego *TCP*

Observa-se o comportamento esperado do *software Alternate Queueing - ALTQ* para o tráfego *TCP* interceptado e encaminhado para a fila *q_tcp*. Na Figura 6.1

Tabela 6.1: *Throughput onix-diamante*

TCP STREAM TEST to 200.202.154.8 : histogram				
Recv Socket Size bytes	Send Socket Size bytes	Send Message Size bytes	Elapsed Time secs.	Throughput 10^3bits/sec
87380	32768	32768	10.02	7530.40
87380	32768	32768	10.03	7655.17
87380	32768	32768	10.04	7474.78
87380	32768	32768	10.02	7397.73
87380	32768	32768	10.03	7814.51
87380	32768	32768	10.03	7387.29
87380	32768	32768	10.03	7420.37
87380	32768	32768	10.03	7344.55
87380	32768	32768	10.02	7469.31
87380	32768	32768	10.03	7431.08
87380	32768	32768	10.03	7394.12
87380	32768	32768	10.03	7281.12
87380	32768	32768	10.02	7375.50
87380	32768	32768	10.04	7418.00
87380	32768	32768	10.03	7460.06
87380	32768	32768	10.03	7439.12
87380	32768	32768	10.02	7520.31
87380	32768	32768	10.03	7364.32
87380	32768	32768	10.02	7461.44
87380	32768	32768	10.05	7405.56

verifica-se que quando ocorre o incremento da banda (acréscimos no paramêtro *bandwidth*) para fila *q_tcp*, haverá o aumento da vazão até o ponto que o tráfego gerado é menor que a banda disponível e não tem como ocupá-la.

6.2.2 Tráfego *SSH*

Observa-se através da Figura 6.2 o comportamento esperado do *software Alternate Queueing - ALTQ* para o tráfego *SSH* (porta 22) interceptado e encaminhado para a fila *q_ssh*. O aumento de banda implementado (de 5% a cada ciclo do experimento) é diretamente proporcional a vazão (*throughput*). Verifica-se que quanto maior a banda (acréscimos no parâmetro *bandwidth*), maior será a vazão e *vice-versa*. Evidencia-se a possibilidade de controle de bandas que o *Alternate Queueing - ALTQ* permite ao administrador de rede, hipótese do presente trabalho.

Tabela 6.2: Geração de tráfego *SSH* através do *sft*

q_tcp	q_ssh	Throughput(Kb/s	Tempo
95%	5%	39,9	42:10
90%	10%	80,7	20:50
85%	15%	120,6	13:56
80%	20%	161,6	10:24
75%	25%	200,8	08:22
70%	30%	240,1	07:00
65%	35%	283,2	05:56
60%	40%	324,2	05:11
55%	45%	362,7	04:38
50%	50%	400,1	04:12
45%	55%	436,5	03:51
40%	60%	475,6	03:32
35%	65%	509,2	03:18
30%	70%	557,0	03:01
25%	75%	593,1	02:10
20%	80%	630,1	02:40
15%	85%	667,7	02:31
10%	90%	681,2	02:28
5%	95%	685,9	02:27
0%	100%	787,7	02:08
Sem ALTQ		908,3	01:51

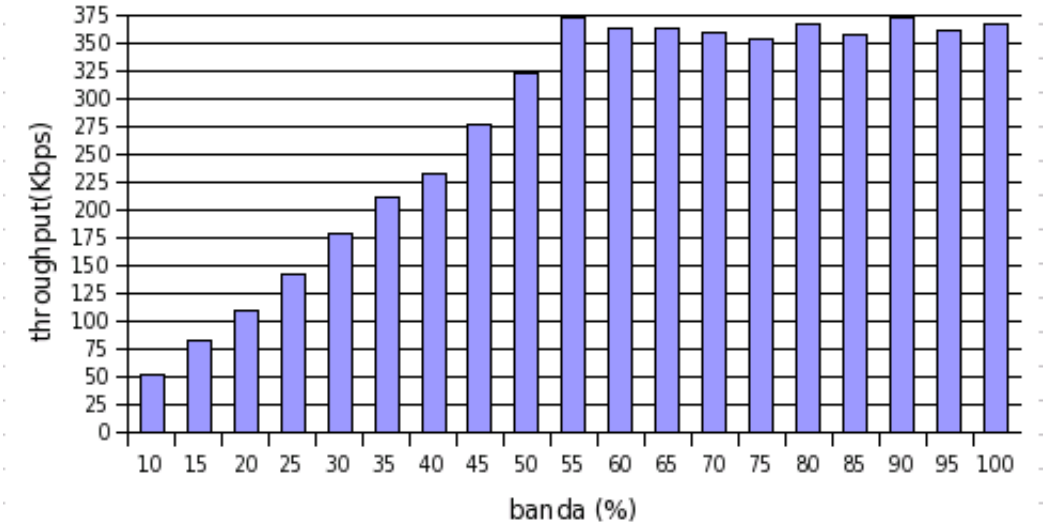

Figura 6.1: Comportamento do tráfego *TCP*

Tabela 6.3: Geração de tráfego *TCP* através do *Netperf*

q_ssh	q_tcp	Throughput(Kb/s
95%	5%	26,19
90%	10%	53,33
85%	15%	84,24
80%	20%	111,27
75%	25%	142,39
70%	30%	180,01
65%	35%	211,28
60%	40%	233,76
55%	45%	276,76
50%	50%	324,54
45%	55%	372,80
40%	60%	365,05
35%	65%	364,71
30%	70%	359,75
25%	75%	353,64
20%	80%	367,13
15%	85%	358,83
10%	90%	374,31
5%	95%	361,46
0%	100%	368,35

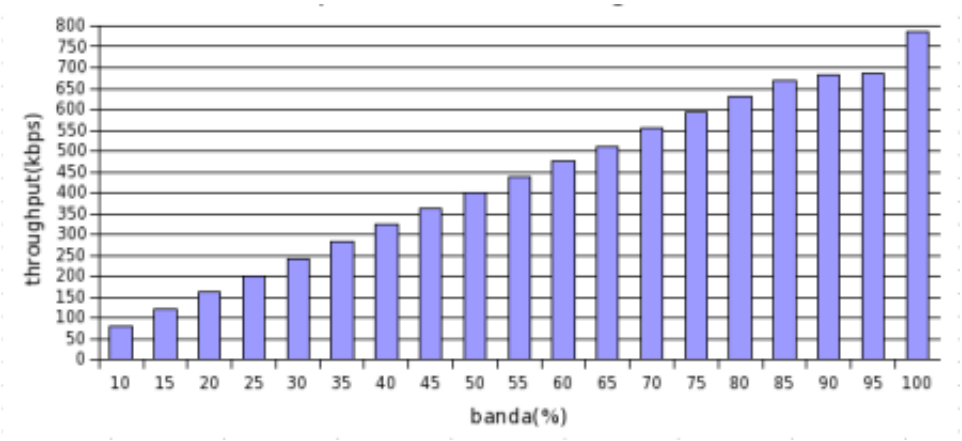

Figura 6.2: Comportamento do tráfego *SSH*

Capítulo 7

Conclusão

Com base na análise dos dados nas seções 6.2.1 - Tráfego *TCP* e 6.2.2 - Tráfego *SSH* e melhor visualisados através dos gráficos (Figura 6.1 - Comportamento do tráfego *TCP* e Figura 6.2) - Comportamentos do tráfego *SSH*, verificou-se a efetividade do *Alternate Queueing - ALTQ* como uma ferramenta de intervenção e controle de bandas de rede.

- A econômia de recursos financeiros na implementação;
- A simplicidade nas operações de definições das filas, prioridades e encaminhamento do tráfego proporcionadas pelo *Packet Filter - PF*;
- A efetividade do *Alternate Queueing - ALTQ* para o controle de bandas de rede.

São fatores que permitem a divulgação deste trabalho e dos resultados para as demais unidades da Embrapa - Empresa Brasileira de Pesquisa Agropecuária, indicando o *Alternate Queueing - ALTQ* como ferramenta de controle de bandas de redes.

7.1 Possibilidades para trabalhos futuros sobre o tema

- O *TC - Trafic Control* é um *software* que tem a mesma funcionalidade do *Alternate Queueing - ALTQ* e executa em sistemas operacionais *GNU-Linux*. Nota-se abundância de casos de usos e documentações sobre os principais *softwares* componentes das distribuições *GNU-Linux*, o que não ocorre com o *TC - Traffic Control*;

- Possibilidade de usos do *Alternate Queueing - ALTQ* como ferramenta de segurança de sistemas;

- Aspectos de degradação do sistemas em virtude de inclusão de regras do *Alternate Queueing - ALTQ* no arquivo de configuração *pf.conf* do *Packet Filter - PF* definindo-se filas, é tópico que merece ser abordado;

Referências Bibliográficas

ANDRADE, R.; KAMIENSKI, C.; SOUSA, D.; SADOCK, D. *O Algoritmo SQM-Response para Controle de Congestionamentos do protocolo TCP*. Brasil, 2003. Disponível em: <www.cin.ufpe.br/˜cak/publications/sbrc2003-sqm-final.pdf>. Acesso em: 18/08/06.

BRADNER, S. *Benchmarking Terminology for Network Interconnection Devices*. IETF, 1991. (Request for Comments). Disponível em: <http://www.ietf.org/rfc-/rfc1242.txt>. Acesso em: 10/07/06.

BROWN, M. A. Site, *Traffic Control HOWTO*. EUA: http://www.tldp.org/Traffic-Control-HOWTO/, 2003. Disponível em: <http://www.tldp.org/Traffic-Control-HOWTO/>. Acesso em: 07/04/06.

CHO, K. *Managing traffic with ALTQ*. Japão, 1999. Disponível em: <ftp://ftp.csl.sony.co.jp/pub/kjc/papers/tm99.ps.gz>. Acesso em: 15/01/06.

CHO, K. *The Design and Implementation of the ALTQ Traffic Management System*. Keio: Keio University, 2001. Disponível em: <ftp://ftp.csl.sony.co.jp/pub/kjc/papers/dissertation.ps.gz>. Acesso em: 13/01/06.

CORREIA, L. H. A.; SILVA, R. M. A. *Redes de Computadores*. Lavras: Ufla-Faepe, 2002.

FARIA, J.; SANTOS, A. *Avaliação de QoS numa arquitectura diffserv*. 2005. Disponível em: <http://www.citebase.org/abstract?id=oai:repositorium.sdum-.uminho.pt:1822/2729>. Acesso em: 29/08/06.

FLOYD, S.; JACOBSON, V. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1993. Disponível em: <citeseer.ist.psu.edu/article/floyd93random.html>. Acesso em: 28/07/06.

HARTMEIER, D. Site, *Prioritizing empty TCP ACKs with pf and ALTQ*. Suiça: http://www.benzedrine.cx/, 2005. Disponível em: <http://www.benzedrine.cx-/ackpri.html/>. Acesso em: 11/05/06.

ONU, S. D. Site, *Millennium Indicatores Development*. EUA: ONU - Statistics Division, 2005. Disponível em: <http://unstats.un.org/unsd/>. Acesso em: 15/03/06.

OPENBSD, P. Site, *PF : The OpenBSD Packet Filter*. EUA: http://www.openbsd.org/, 2006. Disponível em: <http://www.openbsd-.org/faq/pf/filter.html>. Acesso em: 12/05/06.

OPENBSD, P. Site, *PF : The OpenBSD Packet Filter*. EUA: http://www.openbsd.org/, 2006. Disponível em: <http://www.openbsd-.org/faq/pf/queueing.html>. Acesso em: 13/07/06.

RAMAKRISHMAN, K.; FLOYD, S.; BLACK, D. *The Adition of Explicit Congestion Notification (ECN) to IP*. IETF, 2001. (Request for Comments). Disponível em: <http://www.ietf.org/rfc/rfc3168.txt>. Acesso em: 11/07/06.

STATS, I. W. Site, *Internet World Stats*. Espanha: [s.n.], 2006. Disponível em: <//http://www.internetworldstats.com/>. Acesso em: 13/04/06.

TAKAHASHI, T. *Sociedade da Informação no Brasil*. Brasília: Ministério da Ciência e Tecnologia, 2000. Disponível em: <http://www.socinfo.org.br-/livro_verde/download.htm>. Acesso em: 07/01/06.

TANENBAUM, A. S. *SIstemas Operacionais Modernos*. Amsterdan: Prentice Hall, 1992.

UCHOA, J. Q. *Segurança Computacional*. 2. ed. Lavras: UFLA-FAEPE, 2005.

UCHOA, J. Q.; SICA, F. C.; SIMEONE, L. E. *Admistração de Redes Linux*. Lavras: UFLA-FAEPE, 2003.

WIKIPEDIA. Site, *IPv4*. Wikipedia, 2006. Disponível em: <http://pt.wikipedia-.org/wiki/Ipv6>. Acesso em: 29/08/06.

Apêndice A

Arquivos de configuração e scripts

A.1 /etc/pf.conf

O arquivo de configuração “/etc/pf.conf”:

```
#	$FreeBSD: src/etc/pf.conf,v 1.1.2.1 2004/09/17 18:27:14 $
#	$OpenBSD: pf.conf,v 1.21 2003/09/02 20:38:44 david Exp $
int_if=“rl0” # replace with actual internal interface name i.e., dc1
ext_if=“rl0” # replace with actual internal interface name i.e., dc1
# Queueing: rule-based bandwidth control.
altq on $ext_if bandwidth 7.4Mb cbq queue { q_tcp, q_ssh }
queue q_ssh bandwidth 5% priority 5 cbq(default)
queue q_tcp bandwidth 95% priority 2 cbq
# Filtering: the implicit first two rules are
block in all
block out all
pass in on $ext_if proto tcp from any to $ext_if keep state queue (q_tcp)
pass out on $ext_if proto tcp from any to $ext_if keep state queue (q_tcp)
pass in on $ext_if proto tcp to any port 22 keep state queue (q_ssh)
pass out on $ext_if proto tcp to any port 22 keep state queue (q_ssh)
```

A.2 Scripts

A.2.1 Estimativa de vazão (*throughput*)

O script est_vazao.sh para verificação de vazão (*throughput*) entre os computadores *onix* e *diamante*. Através do *script* gerou-se vinte resultados apresentados em 6.1. Utilizou-se a média dos resultados para definir o total da banda.

```
#!/usr/local/bin/bash
h=$1
ip=200.202.154.8
t=10
fl=vazao_onix_diamante
for ((i=1; $i < 21; i++))
do
/usr/local/netperf/netperf -l $t -H $ip -f k » $fl
done
```

A.2.2 Gerador de tráfego TCP

O *script dispara.sh* gera tráfego TCP entre as máquinas *onix* e *diamante*.Executou-se este *script* na máquina *onix*.

```
#!/usr/local/bin/bash
h=$1
ip=200.202.154.8
t=80
rt=6000
fl=trafego_onix_diamante
for ((i=1; $i < 11; i++))
do
/usr/local/netperf/netperf -l $t -t TCP_RR -H $ip – -r $rt,$rt» $fl
done
```

A.3 Aplicações das diretrizes

Segue uma adaptação de um caso de uso apresentado em (OPENBSD, 2006b).

Supõem-se uma pequena rede com um computador executando o sistema operacional *FreeBSD* com *"Alternate Queue - ALTQ"*, localizado entre a internet e

uma rede local composta por 6 computadores (figura A.1). A conexão com a internet é feita através de uma linha *ADSL* [1] com 5 Mbps para a entrada (*download*) e 1Mbps para a saída (*upload*).As condições de controle de bandas de rede são:

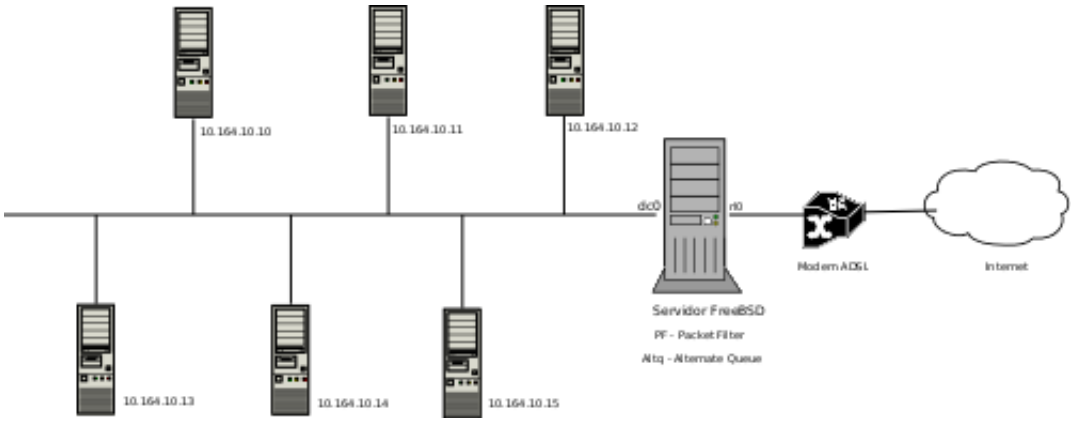

Figura A.1: Esquema da rede

- Os computadores do setor financeiro, cujos endereços *IP* são *10.164.10.10* e *10.164.10.11* deverão ocupar até 2Mbps. O computador 10.164.10.10 (*chefia*) deverá ficar com 70% da banda destinada e o computador 10.164.10.11 (*escriturario*) ficará com a banda restante. Ambos poderão tomar emprestado 2Mps da banda quando houver disponibilidades.
- O *SSH* que permite conexão de usuários com outras máquinas terão prioridades maiores.
- Consultas ao *DNS* [2] - deverão ter a segunda maior prioridade.
- Pacotes *TCP ACK* [3] tenham a maior prioridade dos tráfegos de saída.

[1] *ADSL* é a sigla para Assymmetric Digital Subscriber Line (Linha Digital Assimétrica para Assinante). Trata-se de uma tecnologia que permite a transferência digital de dados em alta velocidade por meio de linhas telefônicas comuns.

[2] *DNS - Domain Name Server* - o servidor de nomes de domínios, segundo (UCHOA; SICA; SIMEONE, 2003), o *DNS* especifica uma estrutura hierárquica de nomes para máquinas e endereços *IPs* em uma forma distribuída. Assim quando uma máquina de um determinado domínio necessita do endereço de uma máquina de outro *site*, há uma troca de informações entre os servidores *DNS* desses domínios

[3] Conforme (HARTMEIER, 2005), quando uma conexão *TCP* é usada para enviar dados somente em uma direção (*downloading* por sftp), *TCP acknowledgements (ACKs)* devem ser enviados em direção contraria para confirmação de chegada das informações (pacotes), caso a confirmação não ocorra haverá retransmissão. Assim é importante que pacotes *TCP-ACK*

A configuração do pf.conf ficará:

```
altq on rl0 priq bandwidth 4500kb queue { q_padrao,q_ssh,q_dns,q_tcp_ack}
```

A diretiva *altq* habilitará a interface externa (*rl0*) para controle de tráfego em direção à Internet. Usa a disciplina *PQ-Priority Queue* (*priq*) e a largura de banda fica definida em 4,5Mbps, pois conforme (HARTMEIER, 2005) no de uso de conexões *ADSL*, uma parte da banda destina-se a controles do protocolo.

Assim define-se as filas:

- q_padrao - A fila padrão, qualquer tráfego não especificado será enviado para esta fila.
- q_ssh - A fila para o *SSH*
- q_dns - A fila para o *DNS*
- q_tcp_ack - A fila para o *TCP ACK*

A diretiva *queue* para atribuição de tráfego de saída e prioridades segue abaixo:

```
queue q_padrao priq(default)
```
Fila *q_padrao* para qualquer tráfego não específico (*default*) - sem prioridades.

```
queue q_ssh priority 2 priq(red)
```
Fila *q_ssh* com maior prioridade que o tráfego não específico.

```
queue q_dns priority 3
```
Fila *q_dns* para consultas de *DNS* com a segunda maior prioridade.

```
queue q_tcp_ack priority 4
```
Pacotes *TCP ACK* com a maior prioridade dos tráfegos de saída.

A diretiva *altq* abaixo habilitará o controle de tráfego vindo da Internet (*dc0*) e utiliza a disciplina de fila *CBQ* para isto:

```
altq on dc0 cbq bandwidth 5Mb queue { q_padrao_ent, q_ssh_ent, q_dns_ent,
q_comp_A}
```

```
queue q_padrao_ent bandwidth 2Mb cbq (default)
```
Fila *q_padrao_ent* do tráfego não específico de entrada - sem prioridades

```
queue q_ssh_ent bandwidth 800Kb priority 2
```
Fila *q_ssh_ent* para tráfego *SSH* com prioridade maior que o tráfego não específico.

```
queue q_dns_ent bandwidth 200Kb priority 3
```
Fila *q_dns_ent* para consulta de *DNS* com prioridade maior.

```
queue q_comp_fin bandwidth 2Mb
q_comp_fin_chefia bandwidth 70% priority 3 (borrow)
q_comp_fin_escrit bandwidth 30% priority 2 (borrow)
```
Fila *q_comp_fin* para os computadores do financeiro com 2Mbps, com duas subfilas - característica do *CBQ - Class Based Queue*.O computador da *chefia* pertencente a fila q_comp_fim_chefia com a possibilidade de ocupar até 70% dos 2Mbps definidos e o computador do escriturário pertencente a fila q_comp_fin_escrit ocupará o restante. Ambos podem tomar emprestado 2Mbps caso hajam disponibilidades.

Macros para redes, interfaces, computadores:
```
rede_local= 10.164.0.0/24
```
$rede_local passa a ser a referência para a rede local.

```
comp_fin_chefia = "10.168.10.10 "
comp_fin_escrit = "10.168.10.11 "
```
$comp_fin_chefia passa a ser a referência para o computador da chefia do setor financeiro.
$comp_fin_escrit passa a ser a referência para o computador do escriturário do setor financeiro.

```
porta_ssh = "22, 2022 "
```
$porta_ssh passa a ser a referência para as portas do tráfego *ssh*.

```
porta_conv = "5022 "
```
$porta_conv passa a ser a referência da porta para troca de mensagens (*Jabber*).

```
ext_if = "rl0"
```
$ext_if passa a ser a referência para a *interface* externa.

```
int_if = "dc0"
```

$int_if passa a ser a referência para a *interface* interna.

As diretivas de *firewall* e filas destino de tráfego para as filas
```
block in on $ext_if all
```
Bloqueia-se todo o tráfego que entra pela *interface* externa

```
block out on $ext_if all
```
Bloqueia todo o tráqfego que sai pela *interface* externa

```
pass out on $etx_if inet proto tcp from ($ext_if) to any flags S/SA
keep state queue(q_padrao_ent, q_tcp_ack)
```
Permite saída de tráfego *IPv4* pela *interface* externa para os *TCP ACKs*.

```
pass out on $ext_if inet proto  udp icmp  from ($ext_if) to any keep
state
```
Permite saída de tráfego *IPv4* pela *interface* externa para protocolos *udp* e *icmp* para qualquer local

Porta para o *DNS*
```
pass out on $ext_if inet proto  tcp udp  from ($ext_if) to any port
domain keep state queue q_dns
```
Permite saída de tráfego *IPv4* pela *interface* externa para protocolos *udp* e *tcp* para consultas de *DNS*

```
pass out on $ext_if inet proto tcp from ($ext_if) to any port $porta_ssh
flags S/SA keep state queue (q_padrao, q_ssh)
pass out on $ext_if inet proto tcp from ($ext_if) to any port $porta_conv
flags S/SA keep state queue (q_ssh, q_tcp_ack)
```
Permite saída de tráfego *IPv4* pela *interface* externa para troca de mensagens através do *Jabber* protegido pelo protocolo *ssh*.

Entrada:
```
block in on $int_if all
```
Bloqueia todo o tráfego que entra pela *interface* interna

```
pass in on $int_if from $rede_local
```
Permite passagem de todo tráfego através da *interface* interna vindo da rede local

```
block out on $int_if
```
Bloqueia todo o tráfego que sai pela *interface* interna.

```
pass out on $int_if from any to $rede_local
```
Permite passagem de todo o tráfego que sai pela *interface* interna para rede local.

```
pass out on $int_if proto  tcp udp  from any port domain to $rede_local
queue q_dns_ent
```
Atribue o tráfego *DNS* para a fila *q_dns_ent*.

```
pass out on $int_if proto tcp from any port $porta_ssh to $rede_local
queue (q_padrao_ent, q_ssh_ent)
```
Atribue o tráfego *SSH* para as filas *q_padrao e q_ssh_ent*.

```
pass out on $int_if proto tcp from any port $porta_conv to $rede_local
queue q_ssh_ent
```
Atribue o tráfego do *jabber* para a fila *q_ssh_ent*.

```
pass out on $int_if from any to $comp_fim_chefia queue q_comp_fin_chefia
pass out on $int_if from any to $comp_fim_escrit queue q_comp_fin_escrit
```
Atribue o tráfego que passa pela *interface* interna para os computadores do setor financeiro para as filas *q_comp_fin_chefia e q_comp_fin_escrit*.