TESTE DA IMPLEMENTAÇÃO DO ALGORITMO *WEIGHTED FAIR QUEING* DESENVOLVIDA NO ÀMBITO do PROJECTO ALTQ

Gonçalo Quadros, António Alves, Edmundo Monteiro, Fernando Boavida
Departamento de Engenharia Informática, Universidade de Coimbra - Pólo II
3030 COIMBRA
Tel.: +351-39-790000, Fax: +351-39-701266, E-mail: quadros@dei.uc.pt

Sumário
A necessidade de dotar os equipamentos intermediários dos sistemas de comunicação de capacidade de fornecerem qualidade de serviço, exige que seja repensado o algoritmo de disciplinamento de pacotes a utilizar – a disciplina *first come first serve* (FCFS) não é adequada. Uma alternativa vulgarmente referida, e utilizada, é o algoritmo WFQ – *Weighted Fair Queueing.* Este artigo apresenta e analisa os resultados de um conjunto de testes a que foi submetida uma implementação para FreeBSD do WFQ (WFQ/ALTQ).

1. INTRODUÇÃO

Está neste momento em desenvolvimento no LCT[1] um trabalho cujo objectivo final é propor, e testar, um novo modelo de serviços para a Internet. Um mecanismo fundamental para esse modelo é o disciplinador de tráfego IP, uma vez que esse influencia de forma determinante as características dos serviços a disponibilizar.
Várias alternativas à disciplina FCFS foram consideradas para serem utilizadas no modelo referido. Procurávamos um algoritmo que pudesse ser utilizado na plataforma actualmente existente no LCT (FreeBSD/Intel), que fosse suficiente eficaz e simples. A implementação WFQ construída no âmbito do projecto ALTQ [1, 2], apareceu naturalmente num lugar cimeiro das hipóteses em consideração.
No âmbito do processo de escolha, submeteu-se a referida implementação a uma série de testes, com o objectivo de avaliar finamente o seu comportamento. Pretendia-se, designadamente, determinar até que ponto é possível através da sua utilização diferenciar de forma controlada o desempenho fornecido a diferentes classes de tráfego.
Este relatório divulga e analisa os resultados dos testes efectuados à implementação referida da disciplina WFQ, executados para avaliar a sua capacidade para suportar tráfego UDP. O ambiente e as ferramentas de testes utilizadas são apresentadas na secção 2. Os testes, seus resultados e respectiva análise, são apresentados na secção 3. Na secção 4, são resumidas as conclusões deste trabalho e apontadas direcções para o fazer evoluir.

2. AMBIENTE DE TESTE

A figura 1 representa o *testbed* utilizado neste trabalho. Nas máquinas HOST_S1 e HOST_S2 foram gerados fluxos de dados destinados a HOST_D, através de HOST_R. As características mais relevantes dessas máquinas são as seguintes: processador PII Celeron/ 333Mhz, 64MB de memória principal, adaptador de rede Intel Pro100B, sistema operativo FreeBSD 2.2.6 com *patch* ALTQ versão 1.0.1.
Os fluxos de dados utilizados nos testes eram constituídos por pacotes grandes, ou seja, com tamanho

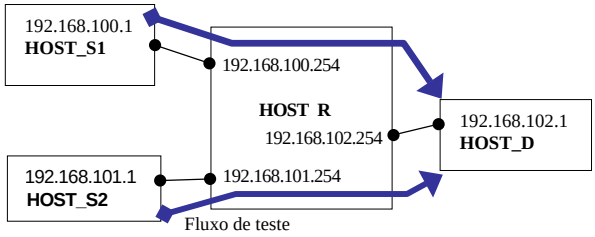

Figura 1 Ambiente de teste

perto do MTU (1500 bytes). Tal não aconteceu por nenhuma razão especial. Por um lado, nesta fase do nosso trabalho, resolvemos fixar esse valor, dado o volume muito grande de testes que teríamos de efectuar se decidíssemos tomá-lo também variável. Por outro, sabíamos que com algumas das ferramentas que estávamos a considerar usar não conseguiríamos gerar fluxos de dados com um débito suficientemente elevado se utilizássemos pacotes pequenos. Foi utilizada a ferramenta *netperf* [3] para a geração dos fluxos de dados.
A ferramenta mais importante na concretização dos testes que aqui se apresentam chama-se *qostat* [4]. Essa ferramenta foi desenvolvida no LCT, e pode ser apresentada como um interface gráfico para a monitorização do processamento IP e das estruturas de dados que essa camada mantém, nomeadamente o que está relacionado com o disciplinamento de pacotes. Com ela é possível monitorar em tempo real o seguinte:
- Valor instantâneo do tamanho da fila de entrada na camada IP (amostragem);
- Valor instantâneo do tamanho de cada uma das filas de saída da camada IP (amostragem);
- Valor médio do tamanho da fila de entrada no nível IP;
- Valor médio do tamanho de cada uma das filas de saída da camada IP;
- Valor do peso associado a cada uma das filas de saída da camada IP (amostragem);
- Valor da quota associada a cada uma das filas de saída da camada IP (amostragem);
- Valor do número de pacotes enviados para o interface de saída (por intervalo de tempo entre amostragens);
- Valor do número de pacotes descartados nas filas de saída (por intervalo de tempo entre amostragens);
- Valor do atraso médio dos pacotes no atravessamento da camada IP.

As medições médias são feitas tendo em conta uma das duas alternativas seguintes:
- Intervalo de tempo entre amostragens (que pode ser configurado para 1, 2, 3, 4 ou 5 segundos);
- Últimos cinco pacotes processados antes do instante da amostragem;

Através da *qostat* é também possível alterar dinamicamente o tamanho máximo das filas de entrada e saída da camada IP (filas ALTQ) e o peso que lhes está associado – o que se revelou uma característica fundamental para a realização dos testes.

[1] Laboratório de Comunicações e Telemática da Universidade de Coimbra.

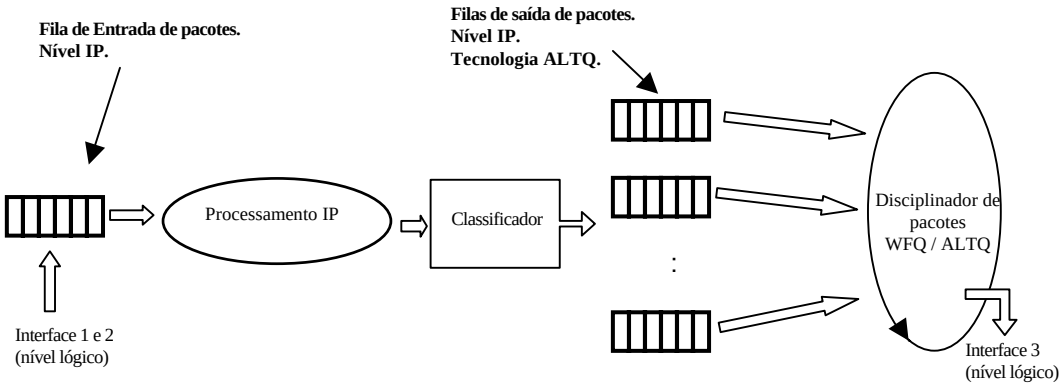

Figura 2 Sistema sujeito ao teste

Para melhor se perceber o que foi monitorizado é apresentado na figura 2 a arquitectura do sistema testado, que corresponde, grosso modo, à camada IP do encaminhador (HOST_R).
As medições referentes ao número de pacotes enviados, número de pacotes descartados e tamanho de filas, referem-se ao interface de saída e à utilização das filas ALTQ aí instaladas. O atraso de trânsito na camada IP é a diferença entre o instante de tempo em que o pacote IP é retirado da fila de saída dessa camada e o instante de tempo em que é inserido na fila de entrada da mesma.
Para terminar a descrição do ambiente de testes, refira-se que foi utilizada como quota base do WFQ o valor configurado por defeito – 512 bytes. Isso significa que, em média, o disciplinador retira em cada visita a uma fila de uma classe configurada com peso 100, 512 bytes.

3. TESTES AO WFQ/ALTQ

Os testes aqui apresentados foram realizados para estudar o comportamento do WFQ, implementação do ALTQ, na presença de tráfego exclusivamente UDP, capaz de esgotar todos os recursos do encaminhador. Mais detalhadamente, para se avaliar a real capacidade desta implementação WFQ fornecer diferentes desempenhos a tráfego pertencente a diferentes classes. Nota-se que a situação de carga mencionada é a mais favorável para que essa capacidade se manifeste.
Os dois fluxos de teste foram atribuídos a duas classes distintas. A uma delas foi dado o peso 100, à outra o peso foi feito variar pelos valores 30, 70, 100, 500 e 1000. Tal só foi possível porque a ferramenta *qostat* permite variar esses valores dinamicamente, ou seja, sem interrupção dos testes.
Detalhando um pouco mais, os testes foram realizados da seguinte forma:
- Os comandos de geração de fluxos foram executados em cada uma da máquinas, de forma a que fossem gerados pacotes à razão máxima possível (perto dos 100Mbps); os fluxos foram atribuídos às classes 1 e 3;
- Após algum tempo, iniciou-se a monitorização utilizando a ferramenta *qostat*. Inicialmente à classe 1 foi atribuído peso 100 e à classe 3 peso 30. A classe 1 manteve sempre o mesmo peso; pelo contrário, o peso da classe 3 foi alterado, depois de cada intervalo de 25 seg, para os valores 70, 100, 500 e 1000. Os resultados são apresentados nas figuras que se seguem[2].

A figura 3 apresenta o atraso de trânsito médio associado ao atravessamento da camada IP. A média é calculada sobre todos os pacotes de cada classe que (para cada intervalo de tempo de um segundo) atravessam a camada IP.
A figura 4 apresenta o número de pacotes de cada classe enviado através do interface de saída por cada intervalo de 1 segundo.
A figura 5 refere-se ao número de pacotes descartados devido a falta de espaço nas filas de saída. Como seria de esperar, e foi verificado nos nossos testes, o descartamento de pacotes aconteceu sempre por falta de espaço nas filas de saída (as filas do ALTQ) e nunca por falta de espaço nas filas de entrada da camada IP.
A figura 6 apresenta o tamanho máximo atingido pelas filas associadas a cada uma das classes em cada intervalo de 1 segundo, enquanto a figura 7 mostra os valores instantâneos do tamanho dessas filas amostrados de segundo a segundo.

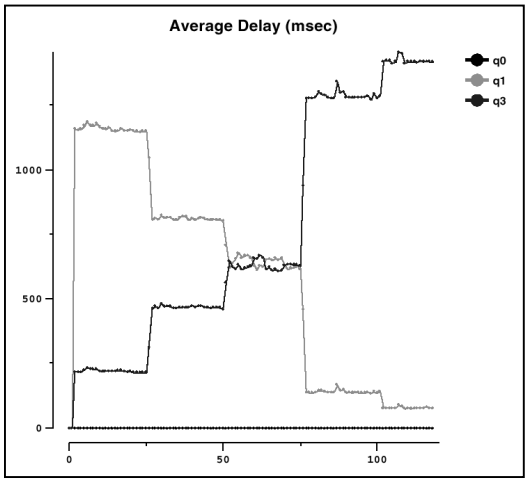

Figura 3
Atraso de trânsito médio dos pacotes na camada IP em cada intervalo de medição de 1s *vs* Peso. Classe 3 começa com peso 30, depois evolui para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75 e 100s. Classe 1 tem sempre peso 100.

[2] Nas figuras aparece representada a classe 0 (ou a fila a ela associada – q0). Tal acontece porque essa classe é configurada por defeito no nosso protótipo. Como não foi gerada tráfego para a classe 0, os valores a elas referentes, para todos os testes, são sempre nulos.

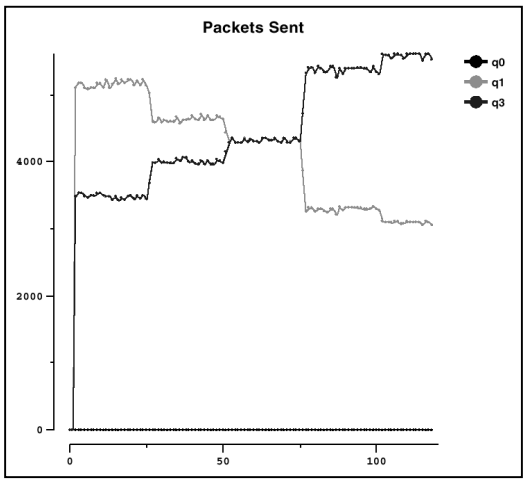

Figura 4

Pacotes enviados por intervalo de medição de 1s *vs* Peso. Classe 3 começa com peso 30, depois passa para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75, e 100 seg. Classe 1 tem sempre peso 100.

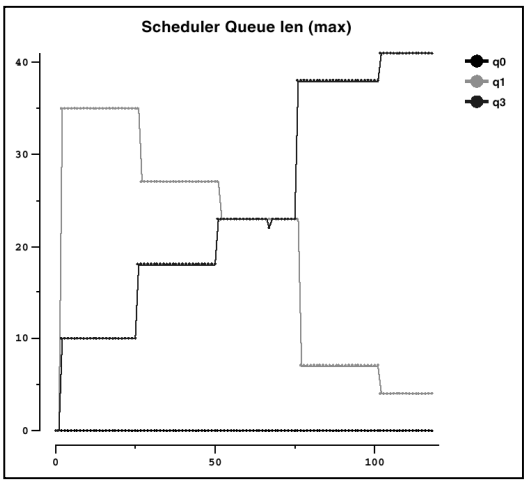

Figura 6

Tamanho máximo atingido pelas filas de saída em cada intervalo de medição de 1s *vs* Peso. Intervalos de medição de 1s. Classe 3 começa com peso 30, depois passa para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75, e 100s. Classe 1 tem sempre peso 100.

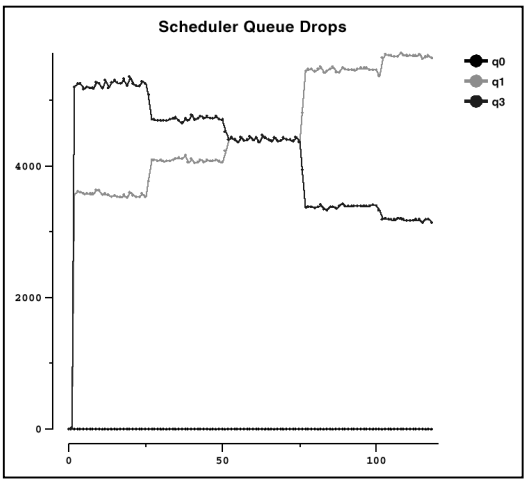

Figura 5

Pacotes descartados por intervalo de medição de 1s *vs* Peso. Classe 3 começa com peso 30, depois passa para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75, e 100 seg. Classe 1 tem sempre peso 100.

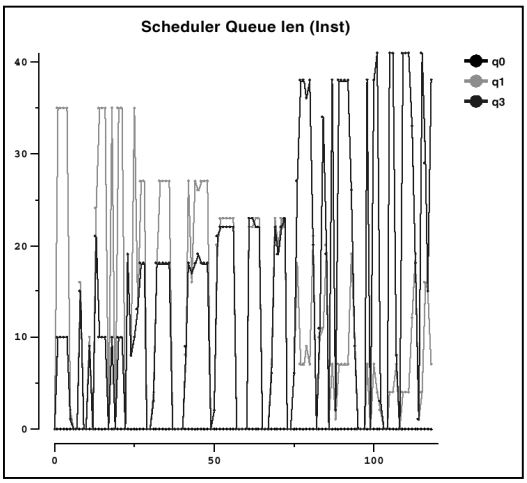

Figura 7

Tamanho das filas de saída amostradas de segundo a segundo *vs* Peso. Valores instantâneos amostrados de segundo a segundo. Classe 3 começa com peso 30, depois passa para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75, e 100s. Classe 1 tem sempre peso 100.

Uma constatação evidente, que importa desde já realçar, é a seguinte: ao contrário do que seria de esperar, os pacotes pertencentes à classe com maior peso são os que demoram mais tempo a atravessar o nível IP. Ou seja, pelo menos no que diz respeito ao atraso de trânsito o comportamento desta implementação não é o desejável.

3.2. Análise dos Resultados

Os resultados dos testes a esta implementação do WFQ revelam alguns factos curiosos. Para além do comportamento inesperado, já referido, do atraso de trânsito, não existe proporcionalidade entre os pesos das classes e o desempenho dado aos seus pacotes. Uma vez que os testes foram executados com carga muito elevada, seria de esperar a existência quase permanente de pacotes em ambas as filas. Assim sendo, se o disciplinador WFQ estivesse a funcionar correctamente, a quantidade de pacotes enviados de cada classe por intervalo de tempo deveria ser proporcional ao seu peso.

No entanto, analisando a Figura 3, podemos verificar que isso não é verdade, tal como também não é verdade para o atraso de trânsito. Decidimos analisar a implementação do algoritmo sujeito ao teste para tentar perceber o que causava tal comportamento. Rapidamente produzimos uma hipótese que nos parecia justificar o observado: era o mecanismo de descartagem utilizado no WFQ/ALTQ o responsável pela diferenciação de tráfego, e não o algoritmo de disciplinamento.

Esse mecanismo faz com que a fila com maior prioridade tenha tendencialmente um tamanho maior do que o da fila com menor prioridade – tanto maior quanto maior for a diferença dos seus pesos. Tal efeito acontece porque a descartagem de pacotes é feita da seguinte maneira: quando é feito o *enqueue* de um pacote numa determinada fila, se o valor do *high water mark* (por defeito 64KB) for ultrapassado, é descartado o pacote que está à cabeça da fila com maior tamanho. O tamanho da fila, no entanto, é calculado tendo em conta o seu peso através da seguinte fórmula:

$$tamanho = bytes\_nos\_pacotes * 100 / peso$$

Suponhamos que estamos a utilizar classes com peso 100 e 1000. Em condições de elevada carga o ritmo de chegada de pacotes a qualquer das filas é bastante alto. Considerando que os pacotes têm todos 1450 bytes, então apenas poderão existir cerca de 45 pacotes em ambas as filas (por imposição do *high water mark*). A fórmula anterior vai provocar que as descartagens se concretizem de tal forma que a fila de maior prioridade tenha cerca de 40 pacotes e a de menor prioridade cerca de 4 pacotes (é com esses valores que os tamanhos, calculados de acordo com a fórmula acima, se igualam).

Se a isso juntarmos uma outra conclusão resultante das pesquisas adicionais que tivemos de efectuar, conseguimos perceber com clareza os resultados obtidos nos testes. Na realidade, algumas experiências extra permitiram-nos ficar a saber que, afinal, apenas cerca de 60 % das vezes em que o disciplinador é chamado para que devolva o próximo pacote IP a processar, encontra pacotes em alguma das filas. Dessas, apenas cerca de 35% das vezes o disciplinador encontra pacotes em ambas as filas.

Em resumo, surpreendentemente, a maioria das vezes em que o disciplinador actua encontra as filas vazias ou apenas uma delas com pacotes. Quando encontra so' uma fila com pacotes, muito provavelmente tratar-se-á da fila com maior prioridade, dado que é essa a maior fila. Ou seja, é dessa fila que sai a maior parte dos pacotes processados, e, por consequência, é da outra fila que são descartados a maior parte dos pacotes em que tal acontece.

Podemos então concluir que a capacidade de diferenciar tráfego não é devida ao funcionamento do mecanismo WFQ mas antes a um efeito co-lateral que provoca uma assimetria forte no tamanho expectável para as filas, resultado do funcionamento do mecanismo de descartagem utilizado. Ou ainda, que é de esperar que quer o atraso de trânsito quer o número de pacotes enviados por intervalo de medição sigam não os pesos dados às classes mas sim o tamanho das filas: quanto maior esse tamanho, maior o atraso e mais provável é o disciplinador encontrar pacotes nessa fila para enviar e portanto maior o número de pacotes enviados por intervalo de medição. A comparação das figura 3, 4 e 6, corroboram esta análise.

Com o objectivo de se verificar inequivocamente que é o mecanismo de descartagem o responsável exclusivo pelo tratamento diferenciado dado aos pacotes das diferentes classes, a ferramenta *qostat* foi modificada para permitir que, dinamicamente, se alterasse o mecanismo de descartagem de tal forma que ele não considerasse os pesos no cálculo dos tamanhos das filas e, consequentemente, não provocasse assimetrias nos tamanhos das mesmas. Os resultados são apresentados nas figuras abaixo e provam que a hipótese que consideramos está certa.

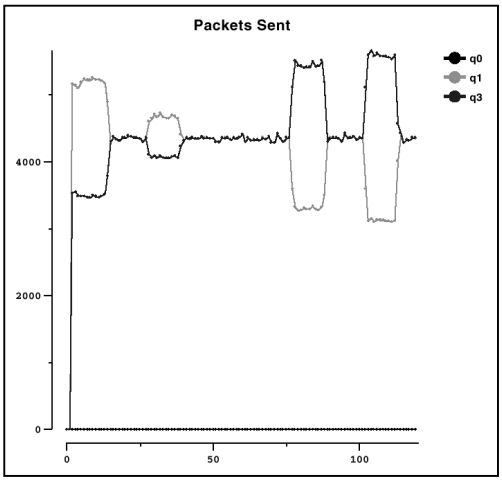

Figura 8

Pacotes enviados por intervalo de medição de 1s *vs* Peso. Classe 1 tem sempre peso 100. Classe 3 começa com peso 30, depois passa para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75, e 100 seg. Na primeira metade de cada um destes intervalos de 25s foi utilizado o peso no cálculo do tamanho das filas. Na segunda metade esse valor não foi utilizado.

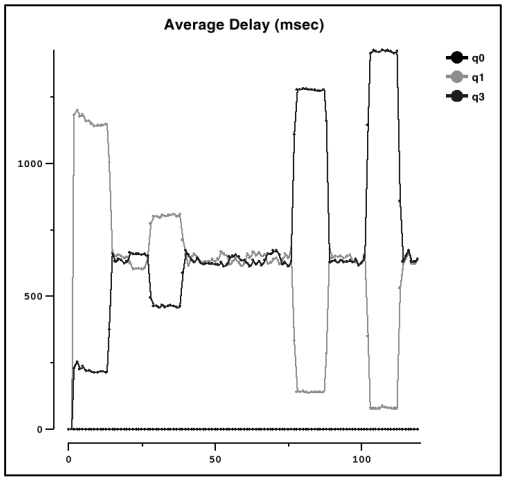

Figura 9

Atraso de trânsito médio dos pacotes na camada IP em cada intervalo de medição de 1s *vs* Peso. Classe 1 tem sempre peso 100. Classe 3 começa com peso 30, depois passa para 70, 100, 500 e 1000, respectivamente aos 25, 50, 75, e 100 seg. Na primeira metade de cada um destes intervalos de 25s foi utilizado o peso no cálculo do tamanho das filas. Na segunda metade esse valor não foi utilizado

4. CONCLUSÃO

A propósito da escolha de um disciplinador para ser utilizado em projectos a decorrer no LCT, resolvemos testar a implementação WFQ do projecto ALTQ (desenvolvido para a plataforma FreeBSD). A sua simplicidade e as diversas referências a este algoritmo, faziam com que fosse, à partida, um candidato forte para ser seleccionado.

Os resultados obtidos nos testes, no entanto, indicam que esta implementação do algoritmo WFQ não funciona como seria de esperar. Mesmo quando utilizado só com fluxos UDP e em situações de carga muito elevada, não é interessante para construção de encaminhadores com capacidade de fornecerem qualidade de serviço.

Na realidade, mostrou-se que a capacidade do WFQ/ALTQ fornecer QoS distinta a diferentes classes de tráfego se deve, única e exclusivamente, ao mecanismo de descartagem de pacotes que utiliza. Esse mecanismo, provoca um efeito preverso: maiores atrasos nas filas com maior prioridade.

Por outro lado, mostrou-se também que com o WFQ/ALTQ não se consegue proporcionalidade entre o tratamento dado aos pacotes das várias classes e o peso que lhes é atribuído. Tal resulta no défice de uma característica importante do disciplinador: a *controlabilidade*. De facto, na utilização que pretendemos dar ao disciplinador (mas não só) é importante a existência de uma relação clara entre os pesos das classes e o desempenho que os seus pacotes podem esperar, por forma a ser possível reagir efectivamente às variações da qualidade de serviço de facto fornecida.

Por estes motivos, este disciplinador deixou de ser considerado para utilização nos projectos actualmente em curso no LCT.

12. REFERÊNCIAS

[1] ALTQ: Alternate Queueing for FreeBSD, www.csl.sony.co.jp/person/kjc/kjc/software.html

[2] Kenjiro Cho, A Framework for Alternate Queueing: Towards Traffic Management by PC Based Routers, in *Proceedings of USENIX 1998 Annual Technical Conference*, New Orleans LA, June 1998. www.csl.sony.co.jp/person/kjc/kjc/papers/usenix98

[3] Hewlett-Packard Company, *Netperf: A Network Performance Benchmark*, Fevereiro 96; (http://www.netperf.org)

[4] Antonio Alves, Goncalo Quadros, *Qostat – Uma Ferramenta de Monitorização de QoS para a Camada IP de um Sistema FreeBSD*, Relatorio Tecnico – CISUC, Junho 99.