Limite de Capacidade e Proteção de Servidores em Redes Gigabit

Marco Antonio Jonack , Cristina Duarte Murta

[1]Universidade Federal do Paraná
Departamento de Informática
Caixa Postal 19081
81531-990 Curitiba, PR

Resumo. *Este trabalho apresenta uma avaliação de dois mecanismos de controle de livelock atuando em diversas máquinas interligadas por uma rede gigabit. O primeiro mecanismo é conhecido como moderação de interrupções e está presente na maioria das interfaces de rede gigabit comercializadas atualmente. O segundo mecanismo, denominado NAPI, é uma nova API para o desenvolvimento de drivers de rede do sistema Linux, que utiliza a técnica de espera ocupada para o tratamento de pacotes. Os resultados mostram que a NAPI é mais eficiente do que a moderação de interrupções em vários aspectos, incluindo estabilidade do sistema em situações de tráfego extremo e utilização intensa de CPU.*

Abstract. *This paper presents the evaluation of two mechanisms for livelock control experienced in computers of different capacities in a gigabit network. The first mechanism is called interrupt moderation. It is implemented in most today high-performance network interfaces. The second mechanism, called NAPI, is a new API for Linux network drivers which uses a polling scheme to handle incoming packets. The results show that NAPI is more efficient than interrupt moderation in many aspects, including the system stability in situation of huge traffic and CPU utilization.*

1. Introdução

Notícias de interesse geral e eventos tais como promoções, datas festivas, acidentes, ataques terroristas, bem como ações mal-intencionadas, como ataques de negação de serviço, podem expor um servidor à uma carga muitas vezes maior do que sua capacidade, caracterizando uma situação de sobrecarga transiente, em que pelo menos um dos seus recursos (CPU, memória, disco, banda de rede ou outro) se torna temporariamente saturado. Este quadro tem motivado a realização de vários estudos e pesquisas relativos ao tratamento e à prevenção de sobrecarga, visto que os métodos tradicionais, tais como Web caching e balanceamento de carga, se mostram pouco eficazes nestes casos [Chen and Mohapatra 2003]. Entre os trabalhos desenvolvidos, o controle de admissão tem se mostrado um mecanismo eficiente e versátil no gerenciamento da sobrecarga em servidores Internet.

Entretanto, a popularização progressiva de redes de alta velocidade, principalmente da ordem de mega e gigabit, que deixam de fazer parte apenas da infra-estrutura da Internet e redes locais para chegar até os usuários e provedores de serviço, muda o foco das atenções para um efeito comum em sistemas orientados à interrupções, o *receive*

livelock [Ramakrishnan 1992], sob o qual até mesmo os mais sofisticados mecanismos de controle de admissão podem se mostrar ineficazes.

Um sistema em *livelock* gasta todos os seus recursos processando interrupções, não dando chance para que outras tarefas, inclusive aquelas relativas a um possível mecanismo de controle de admissão, sejam executadas. Para um observador externo, o sistema parece estar em *deadlock*, pois nenhum avanço é visto no que se refere aos processos do sistema e, conseqüentemente, nenhum progresso em termos de vazão (*throughput*).

Alguns mecanismos para controle de *livelock* foram propostos e implementados nos diversos sistemas operacionais, incluindo BSD [Mogul and Ramakrishnan 1996], Windows NT [Hansen and Jul 1997] e Linux [Salim et al. 2001]. Também é crescente o número de interfaces de rede, principalmente interfaces gigabit, que oferecem algum tipo de mecanismo para limitar o número de interrupções gerado pela chegada de pacotes, o que ajuda a diminuir a carga imposta à CPU do sistema.

Contudo, muitos dos trabalhos anteriores no âmbito de controle de *livelock* conduziram seus testes em redes com velocidades da ordem centenas de megabits por segundo [Druschel and Banga 1996, Mogul and Ramakrishnan 1996, Hansen and Jul 1997, Brustolini et al. 2000, Salim et al. 2001]. Este trabalho apresenta a avaliação de dois mecanismos de controle de *livelock* em uma rede de gigabit. O primeiro mecanismo é conhecido como moderação de interrupções, e está presente na maioria das interfaces de rede gigabit comercializadas atualmente. O segundo mecanismo, denominado NAPI, é uma nova API para o desenvolvimento de *drivers* de rede do sistema Linux, que utiliza a técnica de espera ocupada para o tratamento de pacotes. Em especial, os mecanismos são testados em máquinas de diferentes capacidades.

Este trabalho também avalia o impacto causado por perdas devidas a mecanismos de controle de *livelock*, sendo este tipo de avaliação desconhecido na literatura. Além disso, este trabalho testa a eficácia dos mecanismos de controle de *livelock* sob o ponto de vista de uma aplicação real. Porém, diferente de [Druschel and Banga 1996], que utilizou o servidor Web NCSA em seus testes, este trabalho faz experimentos com o servidor Web Apache. Adicionalmente, os testes feitos neste trabalho complementam a avaliação da NAPI apresentada em [Salim et al. 2001].

Os resultados mostram que a NAPI é mais eficiente do que a moderação de interrupções em vários aspectos, incluindo estabilidade do sistema em situações de tráfego extremo e utilização da CPU, revelando-se um mecanismo de proteção efetivo contra *livelocks*. Também foi constatado que a NAPI produz um grande número de perdas de pacotes, o que certamente não é desejado em sistemas que devem cumprir requisitos de qualidade de serviço.

O restante deste artigo é organizado como descrito a seguir. A Seção 2 discorre sobre os principais mecanismos e técnicas empregadas no controle de sobrecarga em sistemas, apresentando os principais conceitos necessários ao entendimento deste trabalho. A Seção 3 descreve a configuração de experimentos definida para avaliar os mecanismos de controle de livelock, cujos resultados são mostrados na Seção 4. A Seção 5 conclui o trabalho.

2. Controle de Sobrecarga e *Livelock*

A abordagem padrão utilizada por grande parte dos sistemas para controlar a sobrecarga é a *contenção de recursos* [Welsh and Culler 2002]. Nesta abordagem, o acesso aos recursos é limitado por parâmetros definidos pelos administradores do sistema em tempo de compilação ou em arquivos de configuração. Estes parâmetros atuam sobre estruturas internas do sistema, como filas, tamanho dos *buffers*, número máximo de conexões ou usuários simultâneos, dentre outros. Sob estas estruturas, muitas vezes a política utilizada para descartar a carga excedente é a *rejeição de cauda* (*tail drop*), na qual as últimas tarefas a chegar são descartadas caso o limite da estrutura tenha sido atingido. É importante notar que a decisão de qual tarefa deve ser rejeitada pode ter um impacto significativo sobre o desempenho do sistema.

O problema da abordagem de contenção de recursos é seu caráter estático. Em muitos casos, o fato de um dado limite sobre um recurso ter sido alcançado não caracteriza necessariamente um estado real de sobrecarga. Por exemplo, o servidor Web Apache [Apache 2005] limita o número máximo de conexões simultâneas que podem ser atendidas. Quando este número é alcançado, independente do estado do sistema (sobrecarregado ou não), o servidor pára de aceitar novas conexões. Em sistemas nos quais picos e rajadas de carga de diversas intensidades são freqüentes [Crovella and Bestavros 1997], é difícil definir uma configuração estática ótima para parâmetros que limitam o acesso aos recursos [Chen and Mohapatra 2003].

Frente a este quadro, várias abordagens mais eficientes para controlar a sobrecarga foram propostas, dentre as quais se destaca o *controle de admissão*, que se apresenta como um mecanismo robusto e eficiente que procura adaptar, de forma dinâmica, a carga à capacidade do sistema. De forma geral, o objetivo de um mecanismo de controle de admissão é limitar a taxa na qual os pacotes ou requisições enviadas por clientes são aceitas e, conseqüentemente, processadas pelo servidor. A decisão de aceitar ou não o pacote ou requisição em questão é tomada com base em alguma medida de desempenho (ou uma combinação delas) do sistema, ou seja, o ajuste da carga à capacidade do sistema é feito dinamicamente.

Vários mecanismos de controle de admissão para servidores foram propostos na literatura, tais como QGuard [Jamjoom and Reumann 2000], PACERS [Chen et al. 2001], SEDA [Welsh et al. 2001]. Controle de admissão baseado em sessões, proposto em [Cherkasova and Phaal 1999], foi também testado em [Chen and Mohapatra 2003], [Carlström and Rom 2002] e [Voigt and Gunningberg 2001]. Entretanto, independente do mecanismo de controle de admissão e de quão sofisticadas são as técnicas que ele utiliza, ele certamente falhará em proteger o sistema da sobrecarga caso não atente para um efeito comum em redes de alta velocidade, o *receive livelock*, discutido a seguir.

2.1. Controle de *Livelock*

A maioria dos sistemas operacionais modernos baseia-se em uma arquitetura orientada a interrupções para tratar eventos de E/S [Silberschatz and Galvin 1997]. A natureza orientada a interrupções destes sistemas, embora eficiente, pode trazer alguns problemas de escalonamento em situações de sobrecarga, visto que o tratamento de interrupções tem prioridade sobre outras tarefas em execução. Isto é especialmente verdade quando se trata do subsistema de rede.

No esquema clássico de tratamento de pacotes, a chegada de um pacote na interface de rede é sinalizada por uma interrupção de hardware, que tem prioridade sobre as demais tarefas do sistema. Uma vez aceito, o pacote é colocado em uma fila de entrada de pacotes IP para ser processado pelas rotinas específicas de protocolo (UDP, TCP, etc.), executadas no contexto de uma interrupção de software, que também tem prioridade alta de execução, porém menor do que interrupções de hardware. Isto significa que as rotinas de processamento de protocolo e qualquer outro processo de usuário são interrompidas sempre que um novo pacote chega ao sistema. Tal situação implica em uma anomalia no escalonamento de tarefas: caso pacotes de rede cheguem a uma taxa muito alta (maior que a capacidade do sistema), o sistema gastará todo o seu tempo processando interrupções.

Esta situação é denominada *receive livelock*, ou simplesmente *livelock* [Ramakrishnan 1992]: o sistema não está em *deadlock*, porém não apresenta qualquer progresso útil, visto que seus recursos estão sendo consumidos pelo processamento de interrupções. Quando a taxa de chegada diminui suficientemente, o sistema volta a apresentar resposta em termos de vazão. Logo, um estado de *livelock* caracteriza um sistema em sobrecarga no qual o ponto de contenção é a CPU.

Em sistemas bem projetados espera-se que a taxa com que os pacotes são processados e liberados para as aplicações destino, isto é, a vazão do subsistema de rede, acompanhe o tráfego de entrada até um ponto chamado de Taxa de Chegada Máxima Livre de Perdas (TCMLP) e se mantenha relativamente constante a partir deste ponto [Mogul and Ramakrishnan 1996]. Contudo, em sistemas propensos a *livelocks*, cargas acima da TCMLP fazem com que o sistema entre em colapso, elevando rapidamente o tempo de resposta e reduzindo a vazão a zero. A Figura 1, adaptada de [Gribble 2001], ilustra de forma conceitual esta situação.

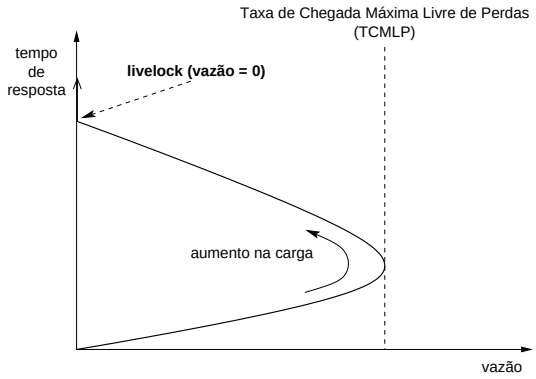


Figura 1. Curva de *livelock*.

O primeiro mecanismo de controle de *livelock* foi implementado para o BSD [Mogul and Ramakrishnan 1996], constituindo-se de uma abordagem híbrida que utiliza tratamento de interrupções e a técnica de espera ocupada (*polling*). Inspiradas nesta implementação, duas outras soluções foram propostas. A primeira [Hansen and Jul 1997] implementa um subsistema de rede para o Windows NT que opera com interrupções e espera ocupada. A segunda [Salim et al. 2001] apresenta uma nova *Application Programming Interface* (API) para o desenvolvimento de *drivers* de rede em Linux, denominada NAPI. Entre os objetivos da NAPI estão a redução do número de interrupções através de tratamento de pacotes, utilizando um esquema misto (interrupções e espera ocupada), e o

descarte antecipado de pacotes, substituindo a fila IP pelo mecanismo de DMA *ring*.

Por outro lado, é crescente o número de interfaces de rede de alta velocidade que implementam mecanismos próprios para moderar a taxa de interrupções, conhecidos como mecanismos de *moderação ou coalescência de interrupções* [Jin and Tierney 2003]. Tais mecanismos podem evitar ou adiar a ocorrência de *livelocks* por limitar a taxa na qual as interrupções são impostas ao sistema. Uma visão mais detalhada sobre questões de desempenho em sistemas computacionais e controle de sobrecarga pode ser encontrada em [Jonack 2005].

3. Configuração dos Experimentos

O objetivo principal dos experimentos é avaliar o desempenho de dois mecanismos de controle de *livelock* atuando em máquinas de diferentes capacidades em uma rede gigabit. O primeiro deles é conhecido como *moderação de interrupções* e está presente na maioria das interfaces de rede gigabit comercializadas atualmente. O segundo, denominado NAPI, é uma nova API para o desenvolvimento de *drivers* de rede do sistema Linux que utiliza a técnica de espera ocupada para o tratamento de pacotes. Para esta avaliação, foi escolhida a interface de rede Intel PRO/1000 modelo MT, comumente chamada de "e1000", cujo *driver* implementa ambos os mecanismos.

Todos os experimentos utilizam a configuração padrão dos mecanismos de controle de *livelock*. Para a NAPI, tais valores são 64 e 300 pacotes, que representam, respectivamente, o número de pacotes processados pelo laço de espera ocupada e pelo subsistema de rede. Para a moderação de interrupções, o *driver* da interface e1000, ajusta a taxa máxima de interrupções e o número de descritores de pacotes de entrada para 8000 interrupções por segundo e 128 descritores, respectivamente. Experimentos indicam que bons valores da taxa máxima de interrupções para sistemas Linux estão entre 1000 e 8000 interrupções por segundo [Intel 2003], o que motiva a utilização do valor padrão do mecanismo de moderação de interrupções.

As próximas subseções detalham o ambiente de execução dos testes, a geração de carga e os testes realizados.

3.1. Ambiente de Testes e Geração de Carga

O ambiente de testes é composto por cinco máquinas interligadas entre si através de um *switch* gigabit, conforme representado na Figura 2. Três máquinas, rotuladas LG_1, LG_2 e LG_3, são responsáveis pela geração de carga. Uma máquina denominada SS_i interpreta o papel de um servidor a ser saturado pelo tráfego das máquinas LG, sendo variada ao longo dos testes. Uma última máquina, nomeada AL, tem a função de analisar o tráfego produzido por SS_i.

A configuração das máquinas geradoras de carga é mostrada na Tabela 1, e a configuração das máquinas que assumem o posto de SS_i (três ao todo) é mostrada na Tabela 2. Em especial, as máquinas da categoria SS_i possuem duas interfaces de rede: uma usada exclusivamente para a comunicação com as máquinas LG (eth0), esta sendo uma e1000, e outra usada para a comunicação com a máquina AL (eth1). Todas as máquinas executam o sistema operacional Linux Debian (kernel 2.6.11.10) e são sincronizadas com o *Network Time Protocol* (NTP) [Mills 1985].

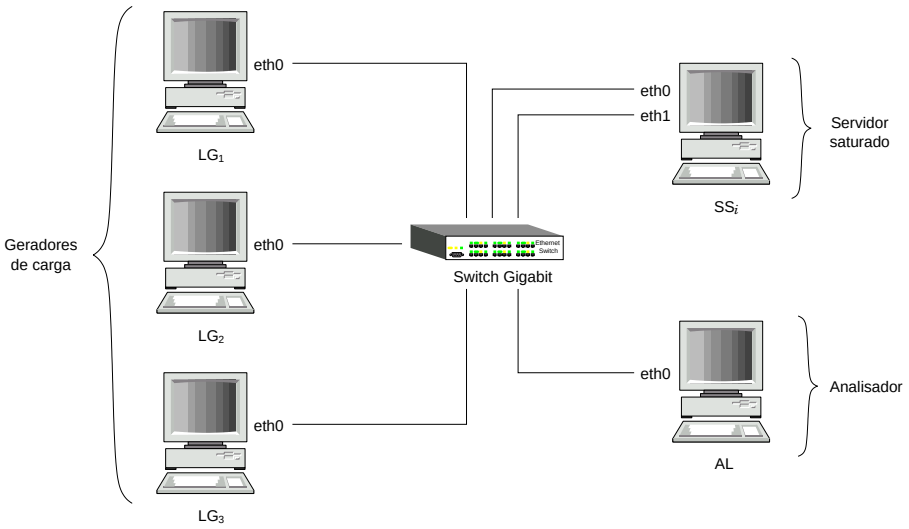


Figura 2. Ambiente de testes.

Tabela 1. Configuração das máquinas LG.

	LG_1	LG_2	LG_3
CPU	Pentium IV 2,8GHz	Pentium IV 2,8GHz	Athlon 1,8GHz
Memória	512MB	512MB	512MB
eth0	e1000	e1000	e1000

A fim de gerar níveis de carga suficientemente altos para os testes realizados neste trabalho, foi escolhida a ferramenta `pktgen` [Olsson 2004], que é executada nas máquinas LG. Entretanto, devido a problemas de configuração, o `pktgen` pode gerar apenas duas taxas por máquina: uma taxa máxima (a máquina envia pacotes o mais rápido que ela pode) e uma taxa intermediária. A combinação das taxas produzidas pela ferramenta `pktgen` em cada máquina LG permitiu a definição de oito níveis de carga (N_c), conforme mostrado na Tabela 3. Cada nível de carga representa um fluxo de pacotes UDP de 64 bytes, transmitidos pelas interfaces eth0 das máquinas LG e recebidos pela interface eth0 de SS_i. Tal fluxo é denotado $(LG, eth0) \rightarrow (SS_i, eth0)$.

Embora seja mais fácil atingir taxas de transferência altas com o uso de pacotes grandes, no caso do fluxo $(LG, eth0) \rightarrow (SS_i, eth0)$ pacotes pequenos são os mais indicados, pois quanto maior a taxa de envio, maior será o número de interrupções causadas pelo recebimento de pacotes em SS_i se comparado a pacotes grandes.

Tabela 2. Configuração das máquinas SS.

	SS_1	SS_2	SS_3
CPU	Duron 750MHz	Athlon 1,2GHz	Pentium IV 3GHz
Memória	256MB	512MB	512MB
eth0	e1000	e1000	e1000
eth1	Tulip 100Mb/s	VIA 100Mb/s	SiS 100Mb/s

Tabela 3. Níveis de carga gerados pelas máquinas LG (Kp/s = Kilo pacotes por segundo).

Nível de Carga (N_c)	Vazão	Máquina(s) Geradora(s)
N_1	127Kp/s ($\sim$ 65Mb/s)	LG_3
N_2	189Kp/s ($\sim$ 97Mb/s)	LG_1
N_3	236Kp/s ($\sim$ 121Mb/s)	LG_3
N_4	324Kp/s ($\sim$ 166Mb/s)	LG_1
N_5	371Kp/s ($\sim$ 190Mb/s)	LG_1, LG_2
N_6	498Kp/s ($\sim$ 255Mb/s)	LG_1, LG_2, LG_3
N_7	639Kp/s ($\sim$ 327Mb/s)	LG_1, LG_2
N_8	875Kp/s ($\sim$ 488Mb/s)	LG_1, LG_2, LG_3

3.2. Testes Realizados

Três tipos de testes foram definidos para avaliar os mecanismos de controle de *livelock*: teste de vazão, teste de utilização de CPU e teste de tempo de resposta. Adicionalmente, foi definido um teste para avaliar o impacto da sobrecarga sobre uma aplicação real, o servidor Web Apache, executado em uma máquina com controle de *livelock*. Cada teste foi repetido para ambos os mecanismos em questão, a saber, NAPI e moderação de interrupções, os quais estão em operação na máquina SS_i.

O teste de vazão avalia a capacidade da máquina SS_i de manter um fluxo constante de 10Mb/s para a máquina AL enquanto é submetida à carga N_c gerada pelas máquinas LG. O fluxo constante, denotado $(SS_i, eth1) \rightarrow (AL, eth0)$, é composto de pacotes UDP de 1470 bytes produzidos pela ferramenta IPerf [NLANR 2005]. O processo IPerf servidor que é executado em AL recebe os pacotes e reporta medições de vazão e *jitter* a cada 0,5 segundo. A vazão de SS_i para a carga N_c é definida como a média dos valores amostrados em AL em um intervalo de 60 segundos, dentro do qual a máquina SS_i estava sendo submetida à carga N_c. O valor da vazão média indicará, de forma indireta, o efeito da carga sobre a máquina SS_i. Neste mesmo teste também são contabilizadas as perdas do fluxo $(LG, eth0) \rightarrow (SS_i, eth0)$, medidas com a ferramenta `netstat` executa antes e após cada teste.

O teste de utilização de CPU mede o impacto da carga N_c sobre a CPU de SS_i. Neste teste, a carga N_c é submetida à máquina SS_i durante 60 segundos. Neste intervalo, a fim de fazer as medições, o comando `vmstat` é configurado em SS_i para imprimir informações sobre o sistema a cada segundo. A utilização de CPU em SS_i para a carga N_c é definida como a média dos valores do campo "cpu/sys" (CPU utilizada pelo sistema) reportados pelo comando `vmstat`.

O teste de tempo de resposta avalia a capacidade do subsistema de rede de SS_i para responder a requisições do comando `ping` enquanto está sob a ação da carga N_c. O comando `ping` é iniciado na máquina AL e configurado de forma a enviar pacotes durante 60 segundos para a máquina SS_i. Ao mesmo tempo, o fluxo $(LG, eth0) \rightarrow (SS_i, eth0)$ é iniciado, o qual também dura 60 segundos (exceto para o primeiro teste, onde o tempo de resposta é medido sem a carga das máquinas LG). A média dos valores de tempo reportados pelo comando `ping` é definida como o tempo médio de resposta de SS_i para a carga N_c.

O último teste tem como objetivo avaliar os efeitos do nível de carga N_c sob uma aplicação real, o servidor Web Apache, que é executado em uma máquina de alta capacidade (SS_3) que opera com um mecanismo de controle de *livelock*. Neste teste, a máquina SS_3 executa a versão 2.0.54 do servidor Web Apache em sua configuração padrão.

A execução de um teste com o servidor Web compreende, além do tráfego produzido pelas máquinas LG, a geração de um fluxo constante de 150 requisições HTTP por segundo entre as máquinas AL e SS_3. Este fluxo, denotado $(AL, eth0) \rightarrow (SS_3, eth1)$, é gerado pela ferramenta `httperf` instalada em AL e simula o acesso a uma página Web estática contendo 15 arquivos, cujos tamanhos variam de 1KB a 5KB. O tamanho médio dos arquivos é 3KB. Esses valores são coerentes com os padrões de acesso observados na Internet [P. Barford and M. Crovella 1998].

A ferramenta `httperf` teve seu código-fonte modificado a fim de reportar, além da vazão, o tempo de resposta a cada 5 segundos. A vazão representa o número de requisições HTTP respondidas a cada segundo (respostas/s). O tempo de resposta corresponde ao tempo entre o envio do primeiro byte da requisição e o recebimento do primeiro byte da resposta do servidor. Os valores destas métricas são definidos como a média das amostras coletadas em AL em um intervalo de 60 segundos dentro do qual a máquina SS_i foi submetida à carga N_c.

4. Resultados

Esta seção apresenta os resultados dos testes que comparam os mecanismos de controle de *livelock* operando em máquinas de diferentes capacidades, bem como os experimentos que avaliam os efeitos de carga sob uma aplicação real, o servidor Web Apache.

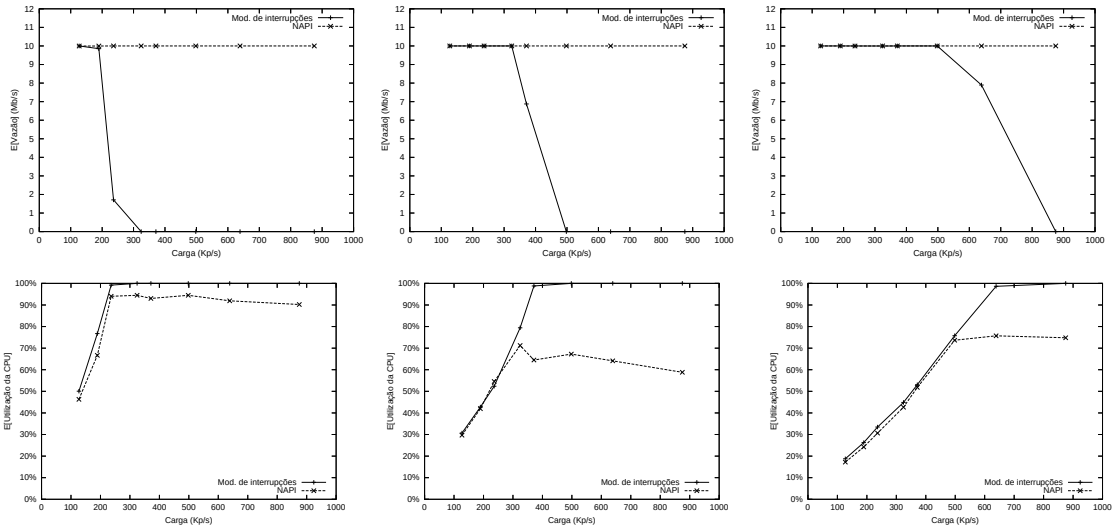
4.1. Moderação de Interrupções *versus* NAPI

A Figura 3 apresenta a vazão do fluxo $(SS_i, eth1) \rightarrow (AL, eth0)$ e a utilização da CPU em função da carga N_c, dada em milhares de pacotes por segundo (Kp/s). Observa-se que as máquinas SS_i utilizando NAPI são capazes de manter a vazão do fluxo $(SS_i, eth1) \rightarrow (AL, eth0)$, independente do nível de carga, enquanto as mesmas máquinas com moderação de interrupções têm a vazão bruscamente diminuída a partir de um certo nível de carga, chegando a zero (*livelock*) no nível seguinte.

Os níveis de carga em que as máquinas SS_1, SS_2 e SS_3 utilizando o mecanismo de moderação de interrupções entram em *livelock*, denotados genericamente como N_l, são N_4 (324Kp/s), N_6 (498Kp/s) e N_8 (875Kp/s), respectivamente. Nestes níveis, inclusive o console das máquinas não responde a comandos e nem mesmo a máquina mais potente (3GHz) é capaz de comportar a taxa de pacotes gerada pelas máquinas LG. Note ainda que, em todos os casos, as máquinas entram *livelock* para cargas que não chegam a utilizar a metade da capacidade nominal da rede.

Também é interessante observar a relação entre a capacidade das máquinas SS_i e o nível de carga no qual elas entram em *livelock* utilizando o mecanismo de moderação de interrupções. Por exemplo, enquanto a máquina de menor capacidade, SS_1 (750MHz), é completamente saturada por um carga de 324Kp/s, é necessário um pouco menos do que o triplo desta carga para saturar a máquina de maior capacidade, SS_3 (3GHz).

Os resultados para a utilização de CPU ajudam a explicar o comportamento da vazão de SS_i. Quando o mecanismo de moderação de interrupções é utilizado, obser-



(a) SS_1 (Duron 750MHz) (b) SS_2 (Athlon 1,2GHz) (c) SS_3 (PentiumIV 3GHz)

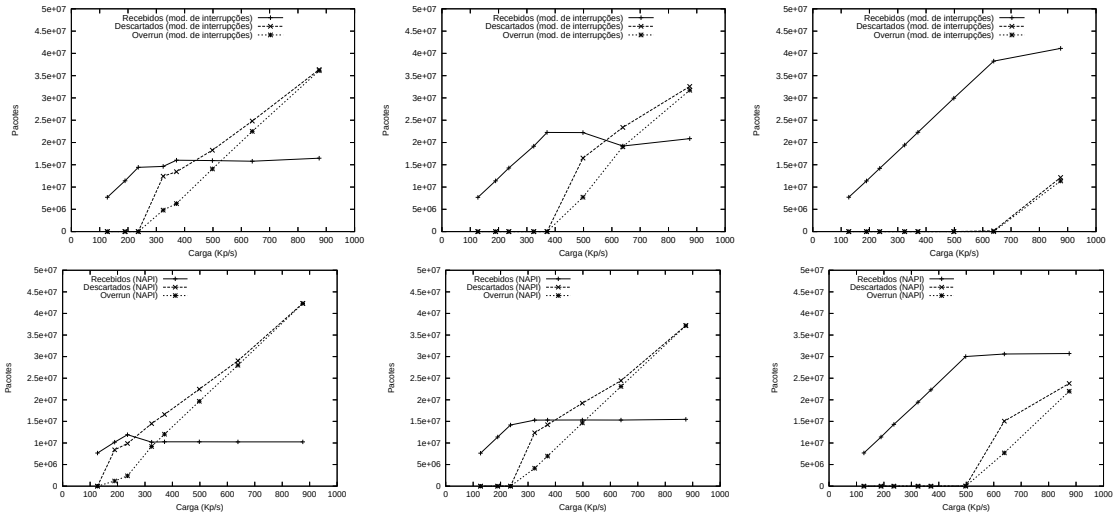
Figura 3. Vazão (gráficos acima) do fluxo $(SS_i, eth1) \rightarrow (AL, eth0)$ e utilização de CPU (gráficos abaixo) em função da carga N_c (milhares de pacotes por segundo) para as três configurações de máquinas SS_i.

vamos que um pouco antes da máquina SS_i entrar em *livelock* a vazão já está muito deteriorada, visto que a utilização da CPU se aproxima de 100%. O mesmo não acontece quando as máquinas SS_i utilizam a NAPI. Ainda assim, a porcentagem de CPU utilizada por SS_1 operando com NAPI é muito alta se comparada às outras máquinas.

Um exame específico das Figuras 3(a), 3(b) e 3(c) revela que a métrica utilização de CPU no caso da NAPI apresenta o mesmo comportamento qualitativo à medida que a carga aumenta: a utilização cresce até certo ponto (que depende da capacidade de SS_i) e depois se estabiliza, tendo uma pequena diminuição. Uma possível explicação para esta diminuição pode ser dada pelo próprio funcionamento da NAPI.O modelo de operação da NAPI, em condições de carga intensa, retira gradativamente a prioridade do subsistema de rede, deixando que outras tarefas sejam executadas, diminuindo assim a utilização da CPU por parte do sistema.

O próximo conjunto de resultados apresenta a quantidade de pacotes recebidos, descartados e *overrun* pela máquina SS_i para cada nível de carga. Neste ponto, algumas considerações sobre o número de pacotes descartados e *overrun* se fazem necessárias. O manual do comando `netstat` descreve os pacotes descartados como o número de pacotes que não puderam ser tratados pelo sistema, possivelmente devido a falta de memória, enquanto os pacotes contados como *overrun* são descritos como o número de pacotes que não puderam ser processados pelo sistema desde a última interrupção, visto que chegaram a uma taxa muito alta. Contudo, tal definição mostra-se insuficiente para uma análise detalhada dos resultados obtidos. Dessa forma, para fins de simplicidade, ambas as quantidades serão referenciadas apenas como *perdas* a partir deste ponto. De fato, uma definição precisa do número de pacotes descartados e *overrun* é muito dependente do hardware subjacente.

A Figura 4 mostra a quantidade de pacotes recebidos e perdas do fluxo



(a) SS_1 (Duron 750MHz) (b) SS_2 (Athlon 1,2GHz) (c) SS_3 (Pentium IV 3GHz)

Figura 4. Número de pacotes recebidos, descartados e *overrun* para testes com moderação de interrupções (gráficos acima) e NAPI (gráficos abaixo) em função da carga N_c.

$(LG, eth0) \rightarrow (SS_i, eth0)$. Da mesma forma que os resultados para as outras métricas, estes também apresentam características qualitativas muito semelhantes: as máquinas recebem pacotes até certo nível de carga, onde então começam as perdas; após este nível, o número de pacotes recebidos tende a se estabilizar, enquanto as perdas aumentam continuamente. Em especial, o comportamento das curvas para a NAPI mostra que este mecanismo estabelece, a partir de um certo nível de carga, um limite superior para o número de pacotes recebidos.

Definindo N_p como o nível de carga em que a máquina SS_i começa a apresentar perdas, os valores de N_{p-1}, N_p e $\frac{1}{N_{p-1}}$ para a máquinas SS_i operando com moderação de interrupções e NAPI são mostrados na Tabela 4. Nesta tabela, N_{p-1} representa o maior nível de carga gerado no qual a máquina SS_i não apresenta perdas, enquanto a fração $\frac{1}{N_{p-1}}$ fornece o tempo entre chegadas de cada pacote para o nível N_{p-1}, o qual pode ser visto como uma estimativa do tempo médio de processamento de pacotes que SS_i pode realizar.

Tabela 4. Valores de N_{p-1} , N_p e $\frac{1}{N_{p-1}}$ para a máquina SS_i operando com moderação de interrupções e NAPI.

Máquinas		N_{p-1}	N_p	$\frac{1}{N_{p-1}}$
SS_1	Mod. de interrupções	236Kp/s	324Kp/s	4,24μs
	NAPI	127Kp/s	189Kp/s	7,87μs
SS_2	Mod. de interrupções	371Kp/s	498Kp/s	2,70μs
	NAPI	236Kp/s	324Kp/s	4,24μs
SS_3	Mod. de interrupções	639Kp/s	875Kp/s	1,56μs
	NAPI	498Kp/s	639Kp/s	2,0μs

A análise dos valores da Tabela 4 mostra que, para as máquinas operando com moderação de interrupções, as perdas começam a acontecer em níveis de carga acima do que as mesmas máquinas utilizando NAPI. Observa-se também que a NAPI insere aumentos significativos no tempo de processamento de pacotes, principalmente para as máquinas mais lentas.

Os resultados descritos acima, se considerados isoladamente, podem sugerir que, em termos quantitativos, a moderação de interrupções é superior a NAPI quando se considera o número de pacotes recebidos e as perdas. Entretanto, nota-se que, utilizando o mecanismo de moderação de interrupções, $N_p = N_l$, ou seja, as perdas só ocorrem quando a máquina entra em *livelock*. Observe também que, segundo os resultados da vazão e utilização de CPU vistos anteriormente, mesmo no nível N_{l-1} o desempenho de SS_i já está bastante comprometido. Logo, o número maior de perdas que a NAPI provoca é justificado e, ao mesmo tempo, evidencia uma de suas principais técnicas, o descarte antecipado de pacotes, como um meio eficiente de proteger o sistema da carga excedente.

Utilizando os resultados obtidos pela análise de perdas, é possível ainda estimar o valor da Taxa de Chegada Máxima Livre de Perdas (TCMLP). Considerando N_t como sendo igual a TCMLP, tem-se que $N_{p-1} \leq N_t < N_p$. Esta formulação é válida para SS_i operando tanto com a NAPI quanto com a moderação de interrupções. Contudo, valendo-se do limite superior definido pela NAPI para o número de pacotes recebidos, é possível calcular um valor aproximado de N_t para SS_i.

Então, supondo que o limite estabelecido pela NAPI não se altera consideravelmente para cargas maiores que N_8, calcula-se o valor de N_t dividindo o número de pacotes recebidos no nível de maior intensidade de carga (N_8) pelo tempo de duração do fluxo $(LG, eth0) \rightarrow (SS_i, eth0)$, que é igual a 60 segundos. Realizando este cálculo, obtém-se os valores aproximados da N_t para as máquinas SS_1, SS_2 e SS_3 como sendo, respectivamente, 171Kp/s ($\sim$ 88Mb/s), 258Kp/s ($\sim$ 132Mb/s) e 512Kp/s ($\sim$ 262 Mb/s). Note ainda que os valores estimados são coerentes com a formulação $N_{p-1} \leq N_t < N_p$ para cada máquina SS_i.

Uma vez que se tenha o valor aproximado da TCMLP, é possível traçar graficamente o comportamento da vazão do subsistema de rede em função da carga de entrada, o que é mostrado pela Figura 5. Nota-se que sistemas que não possuem mecanismos efici-

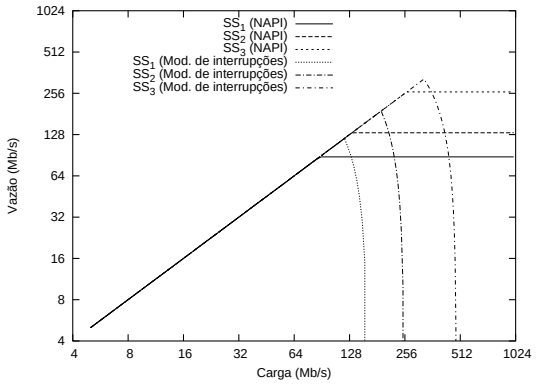
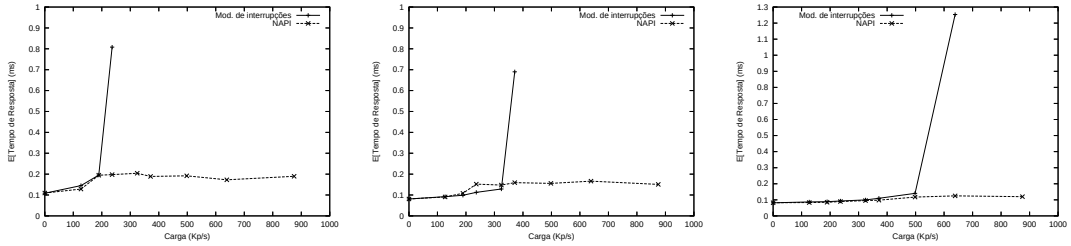


Figura 5. Vazão do subsistema de rede em função da carga de entrada para cada máquina SS_i operando com moderação de interrupções e NAPI.

entes para descartar a carga excedente entram em colapso para cargas além da sua capacidade, o que é provado pelo comportamento das curvas para a moderação de interrupções operando em sua configuração padrão. Por outro lado, as máquinas SS_i com NAPI acompanham a carga até a TCMLP, ponto a partir do qual mantém a vazão constante independente da carga de entrada.

O último conjunto de resultados, exibido na Figura 6, ilustra o comportamento do tempo de resposta em função da carga N_c para cada máquina SS_i. Os gráficos mostram que o tempo de resposta é praticamente idêntico para moderação de interrupções e NAPI durante o primeiros níveis de carga, ficando muito próximo a 0,1ms. Essa situação se mantém apenas até o nível de carga N_{l-1}, quando a curva do tempo de resposta referente à moderação de interrupções apresenta um aumento exponencial. Contraditoriamente, o pior caso desse aumento é observado para a máquina de maior capacidade (máquina SS_3, Figura 6(c)), na qual a diferença entre o menor e o maior valor do tempo de resposta chega a quase a 1,2ms, o que corresponde a um aumento de uma ordem de grandeza.



(a) SS_1 (Duron 750MHz) (b) SS_2 (Athlon 1,2GHz) (c) SS_3 (Pentium IV 3GHz)

Figura 6. Tempo de resposta em função da carga N_c.

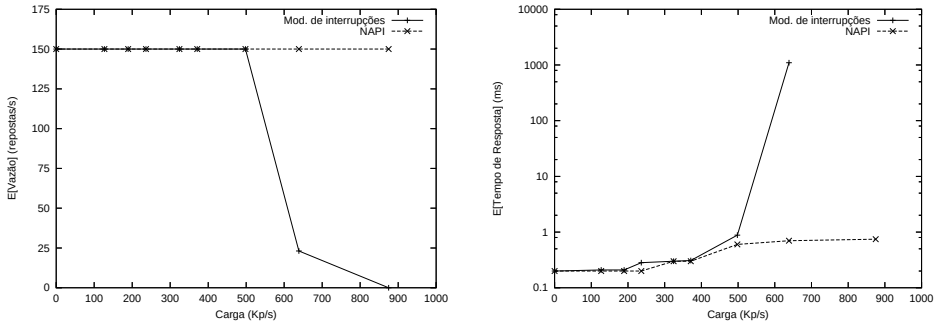
De forma geral, as análises apresentadas nesta seção indicam que a NAPI é mais eficiente do que a moderação de interrupções em vários aspectos, incluindo estabilidade do sistema em situações de tráfego extremo e utilização da CPU. A NAPI também se apresentou mais eficiente em diversas configurações de máquinas, incluindo máquinas de baixa capacidade, sendo capaz de impedir a ocorrência de *livelock* em todos os casos. O mesmo não foi verificado para a moderação de interrupções, que permitiu a ocorrência de *livelocks* em certos níveis de carga.

A análise de perdas, aliada aos resultados da vazão e utilização de CPU, mostrou a eficiência de uma das principais técnicas utilizadas pela NAPI para proteger o sistema da carga, o descarte antecipado de pacotes. Entretanto, a falta de controle de *qual* pacote será descartado pode ser um problema para sistemas que devem implementar características de qualidade de serviço.

4.2. Desempenho do Servidor Web Apache

Conforme descrito na Seção 3, a avaliação dos mecanismos de controle de *livelock* utilizando uma aplicação real, o servidor Web Apache, foi feito sob o ponto de vista de duas métricas: vazão e tempo de resposta.

A Figura 7 mostra os resultados para as métricas consideradas. É observado que o servidor mantém a vazão constante para a maioria das cargas aplicadas sobre máquina a SS_3. A vazão começa a ser afetada somente a partir do nível de carga N_7 (639Kp/s)



(a) Vazão (respostas/s). (b) Tempo de resposta (ms).

Figura 7. Desempenho do servidor Web Apache para a máquina SS_3 (Pentium IV 3GHz)

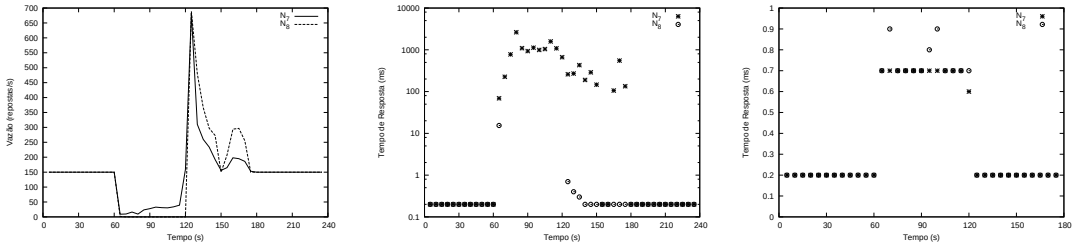
para a máquina SS_3 operando com moderação de interrupções, efeito que é expresso por uma perda de desempenho de aproximadamente 64%, enquanto o tempo de resposta (Figura 7(b)) aumenta praticamente em 1000 vezes. No nível N_8, a taxa de resposta da máquina SS_3 chega a zero quando o mecanismo de moderação de interrupções é utilizado, indicando o estado de *livelock*, fato que também é indicado pela ausência de um valor para o tempo de resposta para este nível.

Os efeitos da carga sobre o desempenho do servidor quando a NAPI está sendo utilizada não são evidentes quando apenas a vazão é considerada. Logo, para esta avaliação, o tempo de resposta é o mais indicado. A Figura 7(b) mostra que o tráfego gerado pelas máquinas LG começa a influenciar o tempo de resposta a partir do nível de carga N_3 (236Kp/s), causando um aumento de 0,2ms para 0,3ms. A partir deste ponto, acréscimos mais significativos são percebidos apenas para os três maiores níveis de carga, nos quais o tempo de resposta chega a 0,7ms.

A Figura 8 exibe os valores individuais da vazão e do tempo de resposta do servidor Web Apache (reportados pela ferramenta `httperf` em AL a cada cinco segundos) ao longo do tempo, para os testes com os níveis de carga N_7 (639Kp/s) e N_8 (875Kp/s), ambos atuando no intervalo [60s..120s] dentro do tempo total de duração do fluxo $(AL, eth0) \rightarrow (SS_3, eth1)$. Nestes testes, a máquina SS_3 utiliza o mecanismo de moderação de interrupções nas Figuras 8(a) e 8(b) e NAPI na Figura 8(c). A vazão para a NAPI não foi considerada visto que se manteve constante durante toda duração do teste.

Nota-se que o servidor Web Apache, executado em SS_3 com moderação de interrupções, não mostra qualquer dificuldade para manter a taxa de resposta constante enquanto não há a presença do tráfego provindo das máquinas LG. Contudo, a partir do instante em que o fluxo $(LG, eth0) \rightarrow (SS_i, eth0)$ é iniciado (segundo 60), a situação muda. No caso do nível de carga N_7, a vazão do servidor cai radicalmente, porém não chega a zero, apresentando um ligeiro crescimento ao longo dos 60 segundos seguintes. O mesmo não acontece para o nível de carga N_8, o qual derruba a vazão a zero (*livelock*), mantendo-a neste patamar durante toda a sua duração. O tempo de resposta aumenta consideravelmente no período de sobrecarga, inclusive para máquina operando com NAPI.

Um comportamento interessante da vazão do servidor, operando com moderação de interrupções, é percebido quando o tráfego das máquinas LG cessa (Figura 8(a), se-



(a) Vazão (moderação de interrupções).

(b) Tempo de resposta (moderação de interrupções).

(c) Tempo de resposta (NAPI).

Figura 8. Valores individuais das métricas do servidor Web Apache ao longo do tempo para os testes com os níveis de carga N_7 (639Kp/s) e N_8 (875Kp/s), ambos atuando no intervalo [60s..120s] dentro do tempo total de duração do fluxo $(AL, eth0) \rightarrow (SS_3, eth1)$. A máquina em questão é SS_3 (Pentium IV 3GHz), a qual opera com o mecanismo de moderação de interrupções (Figuras 8(a) e 8(b)) e NAPI (Figura 8(c)).

gundo 120). A partir deste instante, a vazão apresenta um súbito aumento, atingindo um valor cerca de quatro vezes maior do que a vazão inicial. O primeiro pico de vazão é acompanhado por um segundo pico de menor intensidade. O mesmo se observa para o tempo de resposta (Figura 8(b)), que demora alguns segundos para voltar ao regime de estabilidade.

Este comportamento ocorre porque o processo do servidor Apache não consegue tratar as várias requisições que chegam, o que implica na lotação das filas de entrada do subsistema de rede associadas ao servidor. Da mesma forma, as respostas às requisições também se acumulam nas filas de saída. Contudo, quando o tráfego enviado pelas máquinas LG é finalizado, o conteúdo das filas de recepção/transmissão é processado à taxa máxima pelo sistema. Porém, enquanto o servidor está atendendo as tarefas acumuladas no período de sobrecarga, novas tarefas chegam ao sistema, as quais têm que esperar para serem atendidas, causando um novo acúmulo. Esse ciclo é uma possível explicação para os padrões vistos na vazão do servidor.

Observa-se também, para a máquina operando com moderação de interrupções, que o tempo que o sistema permanece instável antes de voltar ao regime normal de operação é equivalente ao tempo de duração da carga gerada pelas máquinas LG, ou seja, 60 segundos. Estes resultados mostram que os efeitos do tráfego excessivo de entrada persistem por um período de tempo após o término deste.

5. Conclusão

Este trabalho apresentou a avaliação de dois mecanismos de controle de *livelock*, conhecidos como moderação de interrupções e NAPI, atuando com suas respectivas configurações padrão. Ambos os mecanismos se propõem a proteger a máquina em que operam de um possível colapso em uma situação de grande tráfego.

Os dois mecanismos foram testados experimentalmente em três máquinas com diferentes capacidades, conectadas em uma rede gigabit. Estas máquinas foram submetidas a diversos níveis de carga. Os resultados mostram que a NAPI é mais eficiente do que a moderação de interrupções em vários aspectos, principalmente no que se refere à estabilidade do sistema em situações de tráfego extremo. Também foi verificado que o

mecanismo de moderação de interrupções não é capaz de evitar o *livelock* nas máquinas consideradas. Mesmo a máquina de maior capacidade (Pentium IV 3GHz) é completamente saturada por uma carga que não chega a utilizar metade da capacidade nominal da rede gigabit.

Os experimentos realizados o servidor Web Apache, executado na máquina de maior capacidade, mostram que, caso se utilize a NAPI, o atendimento de requisições não é consideravelmente afetado mesmo quando a máquina em questão é submetida a uma carga elevada. Além disso, os testes mostraram que os efeitos da sobrecarga sobre um sistema que não opere com um mecanismo de controle adequado persistem mesmo após o término do sobrecarga. Os resultados indicam que o tempo que tal sistema permanece instável após o fim da sobrecarga é proporcional ao tempo de duração do tráfego de entrada que o saturou.

Este artigo apresenta evidências de que qualquer mecanismo de controle do sobrecarga que não atue de forma efetiva no tratamento dos pacotes de rede ou que não utilize um mecanismo adicional de controle de livelock certamente falhará ao enfrentar situações tráfego extremo em redes de alta velocidade.

Referências

Apache (2005). The Apache Web Server. http://www.apache.org.

Brustolini, J., Silberschatz, A., and Singh, A. (2000). Signaled Receiver Processing. In *Proceedings of the 2000 USENIX Annual Technical Conference*.

Carlström, J. and Rom, R. (2002). Application-aware Admission Control and Scheduling in Web Servers. In *Proceedings of the IEEE Infocom 2002 Conference*, pages 506–515.

Chen, H. and Mohapatra, P. (2003). Overload Control in QoS-aware Web Servers. *Computer Networks*, 42(1):119–133.

Chen, H., Mohapatra, P., and Chen, X. (2001). An Admission Control Scheme for Predictable Server Response Time for Web Accesses. In *Proceedings of the 10th international conference on World Wide Web*, pages 545–554.

Cherkasova, L. and Phaal, P. (1999). Session Based Admission Control: A Mechanism for Improving Performance of Commercial Web Sites. In *Proceedings of the 7th International Workshop on Quality of Service*, pages 226–235.

Crovella, M. and Bestavros, A. (1997). Self-similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE/ACM Trans. Networking*, 5(6):835–846.

Druschel, P. and Banga, G. (1996). Lazy Receiver Processing (LRP): a Network Subsystem Architecture for Server Systems. In *Proceedings of the 2nd USENIX Symposium on Operating Systems Design and Implementation*, pages 261–275. ACM Press.

Gribble, S. D. (2001). Robustness in Complex Systems. In *Proceedings of the 8th Workshop on Hot Topics in Operating Systems*.

Hansen, J. D. and Jul, E. (1997). A Scheduling Scheme for Network Saturated NT Multiprocessors. In *Proceedings of the USENIX Windows NT Workshop*.

Intel (2003). Interrupt Moderation Using Intel Gigabit Ethernet Controllers. http://www.intel.com/design/network/applnots/ap450.htm. Application Note (AP-450).

Jamjoom, H. and Reumann, J. (2000). QGuard: Protecting Internet Servers from Overload. Technical Report CSE-TR-427-00, University of Michigan.

Jin, G. and Tierney, B. L. (2003). System Capability Effects on Algorithms for Network Bandwidth Measurement. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*.

Jonack, M. A. (2005). Limites de Capacidade e Proteção de Servidores em Redes Gigabit. Master's thesis, Universidade Federal do Paraná.

Mills, D. (1985). RFC 958 – Network Time Protocol (NTP).

Mogul, J. and Ramakrishnan, K. K. (1996). Eliminating Receive Livelock in an Interrupt-Driven Kernel. In *Proceedings of the 1996 USENIX Technical Conference*.

NLANR (2005). IPerf. http://dast.nlanr.net/Projects/Iperf/.

Olsson, R. (2004). pktgen The Linux Packet Generator. In *Proceedings of the 11th International Linux System Technology Conference*.

P. Barford and M. Crovella (1998). Generating Representative Web Workloads for Network and Server Performance Evaluation. In *Measurement and Modeling of Computer Systems*, pages 151–160.

Ramakrishnan, K. K. (1992). Scheduling Issues for Interfacing to High Speed Networks. In *Proceedings of the Global Telecommunications Conference*.

Salim, J. H., Olsson, R., and Kuznetsov, A. (2001). Beyond Softnet. In *Proceedings of the 5th Annual Linux Showcase & Conference*.

Silberschatz, A. and Galvin, P. B. (1997). *Operating System Concepts*. John Wiley & Sons, 5th edition.

Voigt, T. and Gunningberg, P. (2001). Kernel-based Control of Persistent Web Server Connections. *ACM SIGMETRICS Performance Evaluation Review*, 29(2):20–25.

Welsh, M. and Culler, D. (2002). Overload Management as a Fundamental Service Design Primitive. In *Proceedings of the 10th ACM SIGOPS European Workshop*.

Welsh, M., Culler, D., and Brewer, E. (2001). SEDA: An Architecture for Well-conditioned, Scalable Internet Services. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles*.