FABRÍCIO BENEVENUTO DE SOUZA

UMA ARQUITETURA PARA MONITORAMENTO E MEDIÇÃO DE DESEMPENHO PARA AMBIENTES VIRTUAIS

Belo Horizonte
28 de março de 2006

Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

UMA ARQUITETURA PARA MONITORAMENTO E MEDIÇÃO DE DESEMPENHO PARA AMBIENTES VIRTUAIS

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

FABRÍCIO BENEVENUTO DE SOUZA

Belo Horizonte
28 de março de 2006

Resumo

Ambientes virtuais têm experimentado um renovado interesse por vários motivos, tais como isolamento de aplicações, consolidação de servidores e compartilhamento de recursos. A implantação de um servidor virtual permite a consolidação de múltiplos sistemas operacionais e aplicações em uma única plataforma de hardware, reduzindo o número de servidores de uma empresa, aumentando a utilização de recursos, simplificando a organização de infraestrutura, reduzindo custos de gerenciamento e permitindo a criação de um ambiente capaz de se adaptar a mudanças na carga das aplicações.

Entretanto, antes migrar aplicações de servidores reais para ambientes virtuais é necessário entender como será o comportamento dessas aplicações nesse novo ambiente. O desempenho das aplicações em ambientes virtuais pode diferir consideravelmente do seu desempenho em sistemas reais devido às interações com a camada de software que simula o hardware e com outras máquinas virtuais. Além disso, a configuração de um ambiente virtual tem um impacto significativo no desempenho do sistema. Por exemplo, no ambiente virtual Xen, dispositivos de E/S são executados em uma máquina virtual privilegiada chamada de domínio de *drivers* isolados (IDD). Uma escolha de configuração que afeta criticamente o desempenho do sistema é a alocação de CPUs para várias máquinas virtuais e IDDs em uma máquina multiprocessada. Nesse contexto, entender as fontes do custo da virtualização e quantificar esse custo é crucial para a implantação e configuração de aplicações em ambientes virtuais.

Esta dissertação apresenta duas técnicas para medir e monitorar desempenho em ambientes virtuais. Utilizando essas técnicas, apresentamos uma avaliação de desem-

penho de aplicações executando no ambiente virtual Xen como uma função das características das aplicações e alternativas de configuração. Os resultados mostram que o processamento de E/S no Xen requer de três a cinco vezes mais processamento de CPU no sistema virtual do que no ambiente real. Além disso, mostramos que a taxa de processamento de rede do IDD é limitada pela capacidade de uma única CPU, mesmo com múltiplos processadores. Foi observada uma queda no desempenho das aplicações, que chega a quase 18%, quando máquinas virtuais compartilham as mesmas caches de processadores. Essas observações são importantes para a configuração de ambientes virtuais bem como a implantação de aplicações nesses sistemas.

Abstract

Virtual environments are experiencing a renewed interest due to several reasons, such as isolation, server consolidation and resource partitioning. The deployment of a virtual server allow the consolidation of multiple operational systems and applications on a single hardware platform, reducing the number of servers of an enterprise, increasing resource utilization, simplifying infrastructure organization, reducing management costs and allowing the creation of an environment able to adapt to changes on application workloads.

However, before migrating applications from physical machines to virtualized environments, one needs to be able to understand the behavior of the applications on the new environment. The performance of such applications on virtual environments may differ considerably from their performance on real systems due to interactions with the software layer which simulates the hardware and due to the need of sharing some with other virtual machines. Moreover, the configuration of a virtual environment has a significant impact on system performance. For instance, in the Xen virtual environment, I/O device drivers execute in privileged virtual machines called isolated driver domains (IDDs). A configuration choice that critically affects system performance is the assignment of multiple virtual machines and IDDs to the CPUs in a multiprocessor server. In this context, understanding the sources of virtualization overhead and quantify this overhead is crucial for the deployment and configuration of applications on virtual environments.

This work presents two techniques to measure and monitor performance in virtual

environments. Using these techniques, we provide a performance evaluation of applications executing on the Xen virtual environment as a function of application parameters and configuration choices. Our results show that I/O processing in Xen requires three to five times the CPU capacity as in Linux. Furthermore, we show that the throughput of an IDD is limited by the processing capacity of a single CPU, even with a multiprocessor IDD. We also observed a performance overhead of almost 18% when virtual machines share the same resources, such as processor caches.

Aos meus pais, minha irmã e minha esposa.

Agradecimentos

Agradeço aos meus pais e minha irmã, que souberam cuidar direitinho do caçula e confiaram em meus sonhos. Obrigado por fazerem tudo isso possível. Ao Vô Zeca, à Vó Amélia e a todos os familiares que torceram pelo meu sucesso. Gostaria de agradecer a toda a família da Carol e dizer que é muito bom fazer parte da família de vocês. Aos meus grandes amigos Ti, Chiquinho e Dandan. Obrigado por vocês estarem presentes em todos os momentos importantes da minha vida e que a nossa amizade continue sempre.

Nestes dois anos de mestrado, tenho certeza que levo comigo grandes amizades para a vida toda, especialmente do pessoal do e-speed. Gostaria de agradecer ao Lhg, Robert, Fernando, Fabiano, Tassni, Barra, Flávia, Brut, Pereira, Adrianoc, Leosilva, Raquel, Coutinho, Michele, Marisa, Vanessa, Krusty, Ítalo, Ismael, Diêgo, Diniz, Gms, Adrianov, César, Matheus, Damacedo, Vitorino e Makish. Gostaria de agradecer ao pessoal da minha turma de graduação (grad001). Eu tive muita sorte de ter feito parte de uma turma tão legal quanto esta. Que esta união e amizade nunca se acabe.

A todos que estiveram envolvidos com a maratona de programação da ACM. Certamente, essa foi uma das experiências mais divertidas da graduação, grande parte devido às pessoas que estiveram envolvidas. Foi um prazer ser competidor por três anos e um orgulho ser técnico.

Aos professores Wagner Meira e Dorgival, responsáveis pelo empurrão inicial durante a graduação e, aos meus orientadores, Virgílio e Jussara, pelos incentivos, orientações, conversas, idéias, oportunidades e ensinamentos.

Boa parte dos resultados apresentados nesta dissertação foram obtidos durante os três meses de *internship* na HP em Palo Alto, uma oportunidade incrível que tenho que agradecer imensamente ao Virgílio. Gostaria de agradecer a todas as pessoas envolvidas no *internship* e que tornaram essa experiência no exterior mais divertida e agradável. Em especial, gostaria de agradecer aos meus mentores na HP José Renato Santos, Yoshio Turner e John Janakiraman, aos outros *interns* e ao Ron e à Beth, os donos do *studio* onde morei.

Por fim, agradeço e dedico esta conquista à minha esposa, Carol, por estar presente em todos os momentos importantes. Você acompanhou de perto cada situação difícil ou feliz, aconselhou, me acalmou e torceu para meu sucesso. Devo tudo a você! A minha parte nessa conquista é bem menor que a sua.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Este capítulo tem por objetivo introduzir o conceito de virtualização, suas aplicações e áreas de atuação. Além disso, apresentamos a motivação e principais contribuições do trabalho. Para concluir, apresentamos uma organização dos demais capítulos da dissertação.

1.1 Virtualização

Recentemente, ambientes virtuais têm experimentado um renovado interesse por vários motivos, tais como isolamento de aplicações, consolidação de servidores e compartilhamento de recursos. O surgimento de ambientes virtuais, como o Xen [9] e o Denali [47], máquinas virtuais comerciais [13] e o desenvolvimento de suporte de hardware para virtualização [22] são exemplos deste ressurgimento de interesse pelo assunto.

Virtualização pode ser entendida como uma técnica que divide os recursos do computador em múltiplos ambientes de execução com o objetivo de implantar novas tecnologias, criando um nível de indireção ou uma camada de abstração entre dispositivos físicos e aplicações. No início dos anos 70, virtualização era um tema de muito interesse da indústria e das universidades. A principal motivação da virtualização nessa época era aumentar o nível de compartilhamento e utilização dos recursos computacionais

de *mainframes*. Entretanto, nos anos 80, o custo do hardware caiu consideravelmente, fazendo com que grande parte das necessidades computacionais de uma organização fossem migradas de grandes *mainframes* para um conjunto de micro-computadores. A motivação da virtualização foi desaparecendo e, com ela, grande parte do investimento comercial. A adoção de micro-computadores associada ao uso de redes de computadores nos anos 90 provocou o surgimento de diferentes paradigmas para a distribuição de computação, tais como sistemas cliente-servidor e par-a-par (*peer-to-peer*). Estes novos tipos de sistema trouxeram diversos desafios e problemas incluindo confiança, segurança, aumento no custo e complexidade de administração, espaço para armazenamento de máquinas e consumo de energia.

O recente renascimento do uso de técnicas de virtualização como produtos para máquinas clientes e servidoras está associado à abordagem dessas questões nesse novo cenário da computação. A implantação de um ambiente virtual em um servidor permite a consolidação segura e flexível de múltiplos sistemas operacionais e aplicações em uma única plataforma de hardware. Isto ajuda a reduzir o número de servidores de uma empresa, aumentar a utilização de recursos, simplificar a organização de infraestrutura, além de reduzir custos de gerenciamento. Além disso, um ambiente virtual pode permitir gerenciamento dinâmico dos recursos de hardware de forma a se adaptar a mudanças na carga das aplicações.

Podemos diferenciar três tipos básicos de virtualização conforme a técnica utilizada: virtualização no nível do kernel, a virtualização completa ou pura e a para-virtualização. O primeiro consiste em virtualizar a camada do sistema operacional [43]. Podemos entender este processo como a divisão de um único servidor em várias partições computacionais (*vservers*), que compartilham o mesmo kernel. Em sistemas Unix, esta técnica se assemelha a uma extensão avançada do mecanismo de *chroot*. No caso da virtualização pura, é possível criar máquinas virtuais que executam sistemas operacionais completos, sem nenhuma modificação no sistema operacional. Este tipo de virtualização, também chamada de virtualização pura, torna mais difícil a execução

de aplicações no topo de um ambiente virtual [46]. A máquina virtual e seu sistema operacional possuem privilégios de usuário, o que causa uma grande dificuldade na execução de determinadas operações, degradando o desempenho das aplicações executadas neste ambiente. Alguns sistemas que se destacam nesta área incluem o VMWare [41], o QEMU [35] e o VirtualPC [39] da Microsoft. Na para-virtualização as máquinas virtuais executam sistemas operacionais modificados para uma arquitetura especial. Estas alterações tornam mais fácil diversas operações do sistema virtual, o que contribui para que esta arquitetura alcance um melhor desempenho do que a arquitetura que utiliza virtualização pura. No caso de servidores, a para-virtualização se torna bastante conveniente por criar um ambiente virtual capaz de particionar os recursos do servidor e isolar aplicações com uma perda de desempenho tolerável [17]. Alguns sistemas que utilizam esta arquitetura são o Xen [9], o Denali [47] e o UML [38].

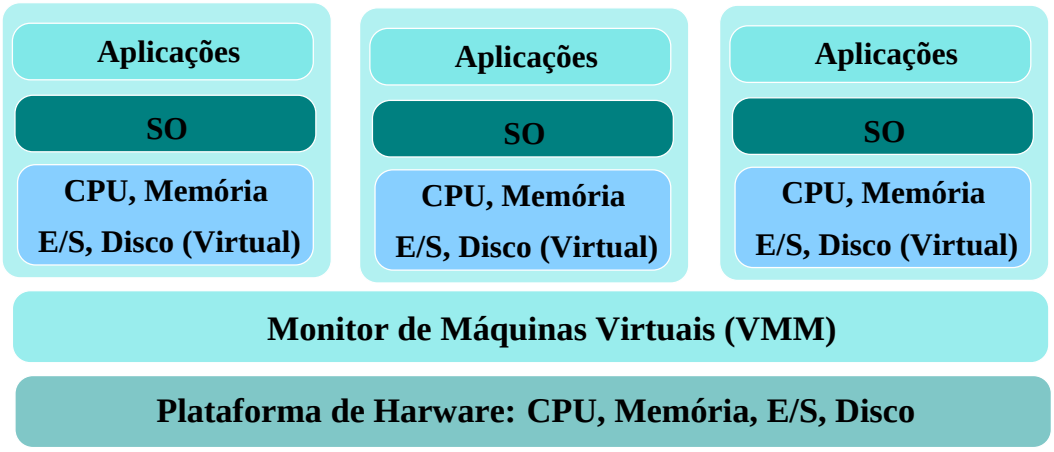


Figura 1.1: Visão geral de um sistema que utiliza para-virtualização

A figura 1.1 mostra uma visão geral de um sistema para-virtual típico. A organização do sistema é realizada através de múltiplas camadas. Na camada mais baixa temos o hardware da máquina física onde o ambiente virtual é executado. No caso de servidores, os recursos são em geral múltiplos processadores, múltiplas interfaces de rede (NICs - *Network Interface Card*), vários discos rígidos e grandes quantidades de memória. O elemento chave para particionar estes recursos em um ambiente virtual está no monitor de máquinas virtuais ou VMM (*virtual machine monitor*). Esta camada

de software possui privilégios de acesso ao hardware para gerenciar as requisições de múltiplos sistemas operacionais hospedados nas máquinas virtuais. Os sistemas operacionais e aplicações executados neste ambiente não sabem que estão em um ambiente compartilhado. Sendo assim, é possível alocar recursos (ex. processamento, memória, E/S, disco) para cada máquina virtual de forma a isolar uma aplicação da outra consolidando diversos serviços em uma mesma plataforma de hardware. O VMM consegue encapsular uma pilha inteira de execução, incluindo o sistema operacional e seu estado atual, de forma que uma máquina virtual pode ser copiada e transferida para outras plataformas ou ambientes.

1.2 Benefícios e Aplicações de Ambientes Virtuais

Várias empresas estão investindo em soluções e serviços que envolvem virtualização. Como exemplo, uma empresa chamada EMC anunciou a compra da VMWare [41], uma das mais famosas empresas que atua na área de sistemas virtuais, por 635 milhões de dólares. Recentemente, a HP lançou um software chamado SoftUDC [23], que consegue agregar, através de virtualização, servidores, redes e sistemas de armazenamento em uma única central de gerenciamento. Outras empresas como a Microsoft e a IBM também estão investindo em suas soluções para ambientes virtuais. Além disso, existe um esforço por parte da Intel para a criação hardware especial para ambientes virtuais para permitir que sistemas operacionais não precisem de modificações para serem executados em ambientes para-virtuais. [22, 14]. Tanto investimento e pesquisa sobre virtualização é devido a um conjunto de vantagens que ambientes virtuais oferecem. A seguir sumarizamos os principais benefícios e aplicações de sistemas virtuais.

- **Consolidação de servidores**: Máquinas virtuais podem ser utilizadas para consolidar vários servidores sub-utilizados em poucas máquinas ou até mesmo em uma única máquina. Isto pode trazer diversas economias para as empresas tanto com a compra de hardware como também com a diminuição de custos

com gerenciamento e administração da infraestrutura dos servidores. Como um exemplo, a VMWare afirma que o custo para a manutenção de servidores pode ser reduzido com o uso de virtualização e consolidação em cerca de 29% a 64% do total de gastos com manutenção [40]. Além disso, a VMWare ainda aponta uma redução de quase 20% em licenças de softwares.

- **Isolamento**: Máquinas virtuais podem isolar o que elas executam, criando um ambiente capaz de conter a propagação de erros e falhas das aplicações e evitar que uma aplicação interfira na execução ou no desempenho de outras aplicações que rodam na mesma máquina.

- **Segurança**: Máquinas virtuais podem ser utilizadas para prover segurança, como por exemplo para criar um ambiente isolado para a execução de aplicações não confiáveis. Empresas que desenvolvem anti-vírus e anti-spyware utilizam virtualização para criar uma rede representativa de máquinas sem segurança conectadas à Internet para poderem monitorar e descobrir novas ameaças na rede [42]. Além disso, um ambiente virtual pode ser utilizado para construir plataformas seguras de computação. Um exemplo disso é o ambiente virtual construído em [4]. Nesse ambiente, diversas aplicações são automaticamente baixadas da Internet e executadas na tentativa de identificar softwares de comportamento malicioso presentes na Web. Nesse caso, o uso de máquinas virtuais oferece não só facilidade e segurança, mas também permite a paralelização do fluxo de execução dos experimentos com o uso de mais de uma máquina virtual.

- **Migração de serviços**: Uma máquina virtual pode ser vista como uma camada de software que encapsula todo o estado do sistema em execução. Isto torna mais simples a migração de aplicações entre máquinas ou processadores, já que o estado da máquina virtual, do sistema operacional e das aplicações podem ser salvos, examinados, modificados e reiniciados. Além disso, virtualização também torna mais simples tarefas como *backup* e recuperação de determinado estado do

sistema.

- **Portabilidade de hardware**: Ambientes virtuais podem ser utilizados para prover a ilusão de determinado hardware, como discos SCSI ou múltiplos processadores. Aplicações que não foram desenvolvidas para determinada plataforma de hardware podem utilizar-se de uma máquina virtual para executarem no hardware adequado. Virtualização também pode ser utilizada para simular redes de computadores independentes em uma mesma máquina ou mesmo executar múltiplos sistemas operacionais diferentes simultaneamente. Além disso, uma máquina virtual pode ser um meio de prover compatibilidade binária, permitindo que aplicações compiladas para uma arquitetura específica sejam executadas em outros tipos de hardware.

- **Ambiente de testes**: Máquinas virtuais fornecem um excelente ambiente para testes, depuração e monitoramento de desempenho. Em um ambiente virtual é possível criar diferentes cenários de testes, como diferentes hardwares, sistemas com recursos limitados e sistemas que utilizam contratos com garantias de qualidade de serviço. Nesse contexto, máquinas virtuais podem ser utilizadas para testar o comportamento de uma determinada aplicação em situações arbitrárias ou criar grandes plataformas de máquinas monitoradas, podendo também ser útil para ambientes de pesquisa e desenvolvimento.

1.3 Motivação do Trabalho

Uma organização típica de TI destina boa parte de seus gastos para gerenciar sistemas existentes e suas aplicações [14]. Uma das fontes deste custo é o grande número de servidores sub-utilizados no centro de dados. Além disso, o desempenho dos servidores aumentou consideravelmente nos últimos anos. Virtualização surge como uma forma de tirar vantagem deste processamento extra para consolidar múltiplas aplicações e sistemas operacionais em uma única plataforma.

Entretanto, antes de migrar aplicações de ambientes não virtuais para ambientes virtuais é necessário entender como será o desempenho dessas aplicações nesse novo ambiente. O desempenho das aplicações em ambientes virtuais pode diferir consideravelmente do seu desempenho em ambientes não virtuais devido às interações com a camada de software que simula o hardware e com as outras máquinas virtuais. Além disso, o custo de desempenho de virtualização pode variar de forma significativa dependendo da configuração do ambiente virtual. Por exemplo, na versão atual da máquina virtual Xen, dispositivos de E/S executam em uma máquina virtual privilegiada chamada de domínio de *drivers* isolados ou IDD (*Isolated Driver Domain*). Uma escolha de configuração que afeta criticamente o desempenho do sistema é a alocação de CPUs a várias máquinas virtuais e IDDs em uma máquina multiprocessada.

Nesse contexto, entender as fontes dos custos de desempenho da virtualização e quantificar este desempenho é crucial para a implantação e configuração de aplicações em ambientes virtuais. Esta dissertação apresenta um levantamento de técnicas para medição e monitoramento de desempenho em ambientes virtuais e, através de um estudo de caso, apresenta uma avaliação de desempenho do ambiente virtual Xen como uma função das características das aplicações e alternativas de configurações.

1.4 Contribuições

Esta sessão sumariza as principais contribuições deste trabalho.

- Apresentamos um levantamento de técnicas para medir e monitorar desempenho em ambientes virtuais. Apesar de enfatizar o ambiente virtual Xen, as técnicas apresentadas podem ser adaptadas para qualquer sistema virtual baseado em para-virtualização.

- Apresentamos um levantamento dos principais aspectos que devem ser considerados ao migrarmos aplicações de um ambiente real para um ambiente virtual.

- Apresentamos uma avaliação de desempenho do ambiente virtual Xen com foco nos parâmetros de configuração que são cruciais para a implantação de aplicações executando em ambientes virtuais. Foram exploradas as principais características do tráfego gerado pelas aplicações que podem ser úteis para alocar recursos e automatizar a configuração de um ambiente virtual. Além disso, apresentamos um estudo sobre a interferência que máquinas virtuais podem causar entre si quando elas compartilham os mesmos recursos como por exemplo as caches do processador. Quando duas ou mais máquinas virtuais compartilham o mesmo processador, uma máquina virtual polui as caches para a próxima máquina virtual, o que pode causar uma degradação no desempenho das aplicações. Como conclusão levantamos novas questões de desempenho ainda não investigadas pela literatura.

- Como um efeito colateral do trabalho, apresentamos uma avaliação de desempenho do Xenoprof [49]. O Xenoprof é uma ferramenta capaz de coletar amostras de eventos de hardware no ambiente virtual Xen. Como utilizamos o Xenoprof para medir desempenho, verificamos o impacto que essa ferramenta pode ter no desempenho do sistema. O Xenoprof foi primeiramente proposto e utilizado em [29], entretanto, aquele trabalho não discute seu impacto no desempenho do sistema.

- Os resultados mostram que o processamento de E/S no Xen requer de três a cinco vezes mais processamento de CPU do sistema virtual quando comparado com o Linux. Além do mais, mostramos que a taxa de processamento de rede do IDD é limitado pela capacidade uma única CPU, mesmo em um IDD com múltiplos processadores. Foi observada uma queda no desempenho das aplicações, que chegou a quase 18% quando máquinas virtuais compartilham as mesmas caches de processadores. Essas observações são importantes para a configuração de ambientes virtuais bem como para a criação de modelos de desempenho para aplicações que executam nestes sistemas.

1.5 Organização do Trabalho

O restante desta dissertação está organizado da seguinte forma. O próximo capítulo apresenta trabalhos relacionados e o capítulo 3 discute as origens da virtualização, aspectos do Xen e do modelo de E/S do Xen, necessários para o entendimento deste trabalho. O capítulo 4 discute técnicas para monitoramento e medição de desempenho de aplicações executadas no Xen. O capítulo 5 apresenta um estudo de caso, onde utilizamos as técnicas discutidas para avaliar o desempenho de um ambiente virtual. Esse capítulo ainda descreve o ambiente experimental utilizado bem como as ferramentas e os conjunto de testes utilizados na avaliação experimental. O capítulo 6 apresenta os resultados do estudo de caso e, finalmente, o capítulo 7 conclui o trabalho e oferece direções para trabalhos futuros.

Capítulo 2

Trabalhos Relacionados

Neste capítulo são discutidos trabalhos relacionados à virtualização, sua evolução e áreas de atuação. Primeiramente, descrevemos trabalhos que deram origem à virtualização e os primeiros ambientes virtuais que surgiram na década de 60. Em seguida, discutimos os principais sistemas responsáveis pelo ressurgimento do interesse em virtualização e discutimos alguns estudos de desempenho existentes sobre ambientes virtuais. Finalmente, discutimos trabalhos sobre *utility computing* e outras áreas que usam como base o conceito de virtualização.

2.1 Evolução da Virtualização

Os estudos e primeiras aplicações de virtualização começaram a surgir em 1960, tendo a IBM como pioneira na área. Naquela época, virtualização era vista como uma evolução dos estudos sobre tempo compartilhado e multiprogramação [6, 36] e se firmou quando a IBM lançou vários projetos que utilizavam máquinas virtuais. Um dos mais populares ambientes virtuais foi o VM/370. O VM/370 é um ambiente virtual que introduziu o conceito de para-virtualização utilizada hoje nos sistemas Xen e Denali. Este foi o primeiro sistema a implementar a idéia da criação de uma máquina virtual de monitoramento ligeiramente modificada em relação ao hardware, que funciona como um sistema operacional com acessos privilegiados, capaz de criar, modificar, migrar,

monitorar e destruir máquinas virtuais. Em relação ao desempenho de máquinas virtuais, Bard realizou estudos usando modelos analíticos baseados em teoria de filas. Em [7] foi desenvolvido um modelo analítico para o VM/370 para estimar medidas de desempenho como utilização de CPU como uma função da carga imposta pelas aplicações executadas no ambiente virtual. Em [8], Bard desenvolveu uma ferramenta para configurar sistemas que utilizam o VM/370 baseada em um modelo analítico que aceita como entrada a carga caracterizada e estima o desempenho do sistema como saída. Recentemente, o estudo de desempenho de ambientes virtuais através de modelagem analítica utilizando teoria de filas foi revisitado [28, 10].

Nos anos 90, virtualização se tornou popular novamente, grande parte devido a uma empresa chamada VMWare [41], que lançou vários ambientes virtuais que dependem ou não de alterações no sistema operacional. Com a retomada do estudo de virtualização e com a evolução do poder de processamento das máquinas atuais e de outros componentes de hardware, surgiram outros projetos e novos sistemas. Dentre eles, dois importantes ambientes virtuais são o Denali [47] e o Xen [9]. A grande diferença entre estes dois sistemas é que os objetivos do Xen são diferentes dos objetivos do Denali. O Denali foi planejado para suportar um grande número de máquinas virtuais, sendo que cada máquina virtual roda apenas uma aplicação. Por outro lado, o Xen foi feito para hospedar máquinas virtuais capazes de suportar sistemas operacionais completos, que podem executar mais de uma aplicação por máquina virtual.

2.2 Desempenho de Sistemas Virtuais

O impacto da degradação no desempenho causado por virtualização foi discutido em vários trabalhos anteriores [9, 25, 47, 46, 44, 37]. Esses trabalhos propõem implementações para máquinas virtuais e avaliam a escalabilidade e o desempenho do sistema de uma maneira comparativa com outros sistemas virtuais. Entretanto, o foco da avaliação de desempenho desses trabalhos está no estudo da eficiência de determinado mecanismo proposto ou comparação com outros sistemas virtuais. Esses trabalhos não

exploram as causas do custo da virtualização, nem estudam o uso da virtualização em um servidor virtual. Neste trabalho, apresentamos uma arquitetura para medição de desempenho visando investigar as causas do processamento de E/S no ambiente virtual Xen, bem como entender o impacto da virtualização em componentes da arquitetura do computador e na interação entre máquinas virtuais.

O foco do estudo de caso e da arquitetura proposta está calcado no modelo atual de E/S do Xen, onde todas as máquinas virtuais se comunicam com o domínio de *drivers* para acessar os dispositivos de hardware [20, 19]. Este modelo é uma tendência crescente e com muitas vantagens, tais como portabilidade, confiança e extensibilidade. De fato, outros trabalhos utilizam técnicas similares às do IDD no Xen [45, 24]. Na seção 3.2 explicamos a arquitetura do Xen para processamento de E/S e seus componentes.

Recentemente, Cherkasova et al. [11] propuseram uma forma de medir o processamento de E/S no IDD devido a uma determinada máquina virtual. A idéia consiste em contabilizar o custo e o número de trocas de páginas entre uma máquina virtual e o IDD para calcular a porção de tempo de CPU do IDD dedicada para cada máquina virtual. Neste trabalho, separamos o processamento do IDD para E/S e o processamento da máquina virtual colocando esses dois componentes do Xen em processadores diferentes. O uso de processadores diferentes para a máquina virtual e o IDD torna desnecessário o cálculo da utilização no IDD, com base no número de trocas de páginas. Em [21], o processamento de CPU para E/S no ambiente virtual Xen também foi estudado. Esse trabalho apresenta uma ferramenta para monitorar a máquina virtual do Xen e quantificar o custo de desempenho. Entretanto, o foco desse trabalho está na distribuição de processamento entre o IDD e a máquina virtual quando ambas competem pela mesma CPU. São explorados diferentes cenários variando parâmetros do escalonador BVT (*Borrowed-virtual-time*).

2.3 Aplicações de Ambientes Virtuais

Um dos principais objetivos dos projetos recentes que envolvem virtualização é alcançar um nível de desempenho que permita a criação de centros de dados com um grande número de máquinas físicas, que podem ser compartilhadas por diferentes unidades administrativas [1, 23]. O motivo desse esforço culminou em um modelo para computação chamado de *Utility Computing*[1]. Nesse modelo os centros de dados de tecnologia da informação executam aplicações de terceiros, que não pagam pelo serviço, mas sim pela quantidade de recursos utilizada [48]. Na direção desse novo modelo de negócios para a computação, várias empresas já possuem sua solução. A HP criou o SoftUDC [23], a Sun possui um sistema chamado N1 ("gerenciar n computadores em 1"). A IBM, assim como várias outras empresas também estão trabalhando nesta área. Alguns esforços mais recentes visam a implantação de modelos de computação por utilidade com qualidade de serviço (QoS) diferenciada de acordo com contratos de nível de serviço (SLA - *Service Level Agreement*) estabelecidos entre clientes e centros computacionais [2, 12].

Outra importante área em que virtualização vem sendo largamente utilizada é área de grades computacionais [18, 26]. A virtualização funciona neste contexto como meio capaz de criar um ambiente flexível, gerenciável e adaptativo, que consiga isolar uma aplicação da outra sem comprometer o desempenho das aplicações. Como um exemplo, o Planetlab [34], que é formado por uma rede de máquinas espalhadas por todo o planeta, implementa virtualização para ampliar seu poder de monitoramento e prover flexibilidade para alocação de recursos.

[1] O termo Utility vem dos modelos adotados para fornecimento de eletricidade, telefone, gás e água. No caso da computação, usuários pagariam pelo processamento de suas aplicações.

Capítulo 3

Sistemas Virtuais

O objetivo deste capítulo é apresentar as informações sobre virtualização e ambientes virtuais necessárias para o entendimento do restante do trabalho. Inicialmente apresentamos uma breve descrição do sistema virtual Xen e do modelo de E/S adotado pelo Xen. Esta descrição está concentrada nos aspectos do Xen que são relevantes para a arquitetura proposta e para o estudo de caso realizado. Em seguida, discutimos os principais pontos que devem ser considerados ao migrarmos aplicações de um servidor real para um ambiente virtual.

3.1 Xen

O Xen é um ambiente virtual de código aberto que permite múltiplas instâncias de sistemas operacionais rodarem concorrentemente em uma única máquina. Este ambiente virtual foi inicialmente desenvolvido para a arquitetura dos processadores x86, mas já existem versões para as arquiteturas de 64 bits. O Xen utiliza o conceito de para-virtualização, onde uma nova camada de software expõe uma abstração de uma máquina virtual, um pouco diferente da camada de hardware. Em geral, para-virtualização alcança um melhor desempenho do que a abordagem baseada em virtualização pura, mesmo em arquiteturas que não foram planejadas para ambientes virtuais [9]. O ponto negativo dessa arquitetura é a necessidade do sistema operacional

ser modificado para a interface da máquina virtual. Aplicações dos usuários e bibliotecas não precisam sofrer modificações. Recentemente, a Intel desenvolveu hardware especial para virtualização para evitar esta necessidade de alteração no sistema operacional. Isso permite, por exemplo, que o sistema operacional Windows seja executado no Xen, sem que a Microsoft lance uma versão modificada do Windows especial para o Xen [14].

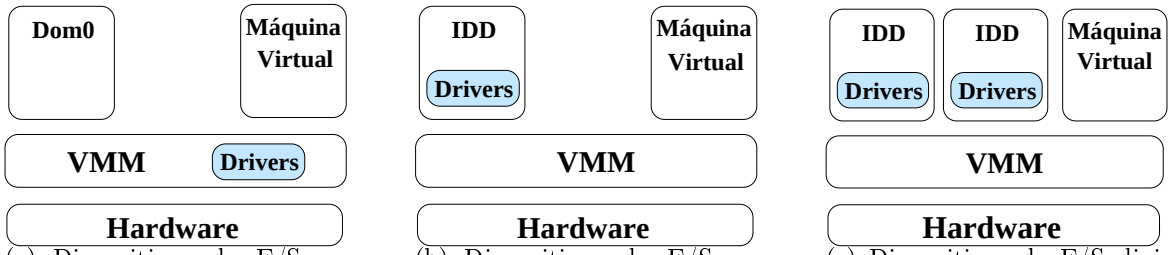
Uma visão geral da arquitetura do ambiente virtual Xen está representada na figura 1.1 do capítulo 1. Como podemos ver, o sistema virtual Xen é organizado em múltiplas camadas. Na camada mais baixa temos o hardware da máquina onde o ambiente virtual é executado. Logo acima da camada de hardware temos a camada de software com mais privilégios de acesso ao hardware chamada de monitor de máquinas virtuais (VMM - *virtual machine monitor*). Acima desta camada se encontram as máquinas virtuais, também chamadas de domínios [9]. O Xen pode hospedar múltiplos sistemas operacionais, cada um rodando em uma máquina virtual. As máquinas virtuais são escalonadas pelo Xen para fazer uso mais eficiente das CPUs disponíveis. Cada sistema operacional gerencia suas próprias aplicações, que muitas vezes precisam de acesso privilegiado ao hardware. Pare este propósito, o Xen expõe um mecanismo de hiperchamada, que as máquinas virtuais precisam utilizar para realizar operações privilegiadas (ex., instalar uma tabela de página), um mecanismo de notificação para entregar interrupções virtuais e notificações para máquinas virtuais e um canal de comunicação para transferir mensagens de E/S entre as máquinas virtuais e o domínio privilegiado.

3.2 Modelo de E/S do Xen

Em um ambiente virtual todas as operações de E/S são processadas diretamente pelo hardware de forma transparente para o sistema operacional das máquinas virtuais.

No primeiro modelo de E/S utilizado pelo Xen, dispositivos de E/S eram mantidos no VMM [9]. A figura 3.1(a) descreve este modelo dentro da arquitetura do Xen. O

VMM possui privilégios de acesso ao hardware, sendo responsável pelo gerenciamento das outras máquinas virtuais. Como todo o tráfego de E/S precisa passar pelo VMM, nesta arquitetura temos problemas de escalabilidade além de um único ponto de falha capaz de parar todo o sistema.



(a) Dispositivos de E/S no VMM

(b) Dispositivos de E/S no IDD

(c) Dispositivos de E/S divididos em múltiplos IDDs

Figura 3.1: Visão geral dos diferentes modelos de E/S do Xen

Recentemente, um novo modelo foi proposto e implementado na arquitetura do Xen [20]. Este modelo utiliza uma máquina virtual com acessos privilegiados chamada de domínio de driver isolado ou IDD (*Isolated Driver Domain*), que contém dispositivos de hardware específicos e executa os dispositivos de entrada e saída. Todas as outras máquinas virtuais executam um único e simples dispositivo de driver que acessa os verdadeiros dispositivos de hardware. A figura 3.2 mostra uma descrição detalhada deste modelo de E/S. O IDD pode acessar diretamente os dispositivos de hardware que ele possui. Por outro lado, uma máquina virtual hóspede[1] utiliza um dispositivo virtual conectado ao IDD para acessar os dispositivos de hardware. O IDD mapeia, através de pontes ou mecanismos de roteamento, a interface física em sua interface virtual que comunica com a interface da máquina virtual como mostra a figura 3.1(b). Interrupções de dispositivos de hardware são primeiramente percebidas pelo VMM, que então notifica o correspondente IDD através de interrupções virtuais entregues por um mecanismo de eventos. As máquinas virtuais trocam serviços de requisições e respostas com o IDD através de um anel descritor de E/S no canal do dispositivo. Referências para páginas são transferidas através do anel de descritores de E/S ao invés

[1] O termo hóspede é usado para designar uma máquina virtual na terminologia do Xen

de enviar os verdadeiros dados, pois isto causaria cópia e duplicação. Quando o dado é enviado pela máquina virtual, o Xen utiliza um mecanismo de compartilhamento onde a máquina virtual permite ao IDD mapear a página com o dado e acessá-la através de acesso direto à memória (DMA - *Dynamic Memory Access*) pelo dispositivo. Quando o dado é enviado pelo IDD para a máquina virtual, o Xen utiliza um mecanismo de remapeamento de página, que mapeia a página fornecida pela máquina virtual. Foi utilizado nos experimentos uma versão do Xen que implementa este modelo.

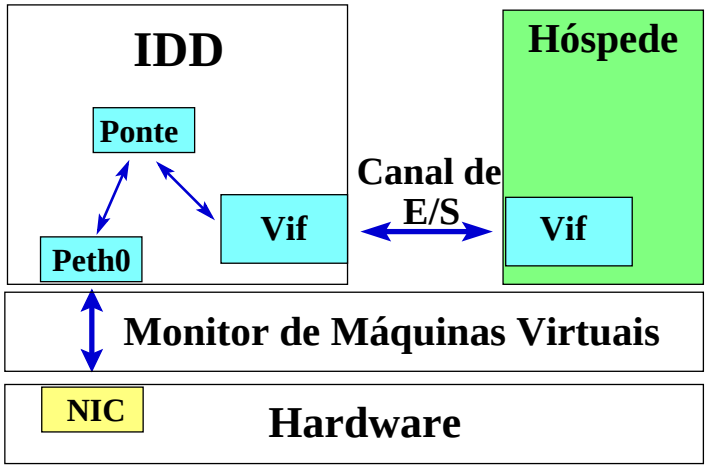


Figura 3.2: Visão geral do modelo de E/S atual do Xen

Outro modelo de E/S recentemente proposto [19], introduz a idéia de múltiplos IDDs. Este terceiro cenário é representado na figura 3.1(c). Espera-se que o suporte a múltiplos IDDs evite alguns problemas relacionados a conflitos de otimizações e alocação de recursos. Além disso, espera-se que esta abordagem possa melhorar a escalabilidade dos domínios de drivers e prover uma melhor tolerância a falhas para o sistema. O suporte para múltiplos IDDs ainda não foi implementado no Xen.

3.3 Implantação de Aplicações em um Ambiente Virtual

A implantação de aplicações em máquinas virtuais levanta vários problemas que não existem em ambientes reais. A seguir discutiremos os principais fatores que devem ser considerados ao migrarmos aplicações para um ambiente virtual. Podemos classificar estes problemas em três grupos básicos:

- **Mecanismos não implementados**: Diversas aplicações utilizam recursos especiais de hardware ou mesmo mecanismos especiais para obter um melhor desempenho. Muitas vezes estes mecanismos não podem ser implantados em um ambiente virtual, o que pode fazer com que determinada aplicação apresente um desempenho bem pior quando comparado com a sua execução em um ambiente não virtual. Por exemplo, a versão atual da virtualização de E/S do Xen pode degradar o desempenho das aplicações por não utilizar o hardware de rede para fazer segmentação TCP e *checksum*. Além disso, aplicações de rede que utilizam a rotina *sendfile()* podem ter uma degradação no desempenho considerável em ambientes virtuais já que esta função consiste de uma otimização para enviar pacotes de rede diretamente de arquivos que não está implementada no Xen. Há um grande esforço sendo realizado para melhorar o desempenho da virtualização, especialmente para E/S [20, 19, 45, 24]. Definitivamente as arquiteturas de hardware de hoje não foram projetadas para ambientes virtuais e o desenvolvimento de hardware especial para virtualização parece ser uma possível direção para contornar diversos problemas, inclusive problemas de desempenho [22].

- **Escalabilidade e Configuração**: O desempenho de aplicações em ambientes virtuais pode variar significativamente dependendo da configuração adotada pelo sistema. Para rodarmos um conjunto de aplicações encapsuladas por máquinas virtuais é crucial determinarmos uma configuração adequada para os recursos físicos disponíveis. Por exemplo, no ambiente virtual Xen, uma escolha de confi-

guração capaz de afetar criticamente o desempenho das aplicações é a associação de múltiplas máquinas virtuais e IDDs aos processadores de um servidor multiprocessado.

- **Interferência entre máquinas virtuais**: Considere um servidor com um número X de processadores rodando um número Y de máquinas virtuais tal que $Y > X$. Nesse cenário, pelo menos um processador será compartilhado por duas máquinas virtuais e, considerando-se que o escalonador do sistema virtual seja justo, poderíamos esperar que o desempenho das aplicações executadas nessa máquina virtual seja próximo da metade do desempenho que elas alcançam quando esta máquina virtual executa em um processador não compartilhado. Entretanto, este desempenho pode ser pior. Máquinas virtuais compartilhando os mesmos recursos de hardware, como caches, podem causar interferência umas nas outras. Uma máquina virtual não é uma simples aplicação. Quando o processador começa a rodar uma máquina virtual, além do código das aplicações em execução, cada máquina virtual carrega todo o código do *kernel* e bibliotecas, poluindo as caches do processador para a próxima máquina virtual. Note que este efeito também acontece em um ambiente real com várias aplicações em execução. Entretanto, as máquinas virtuais são mais "pesadas"e podem causar uma degradação de desempenho maior.

Capítulo 4

Monitoramento e Medição de Desempenho em Servidores Virtuais

Este capítulo apresenta uma discussão sobre medição de desempenho nos vários níveis de um ambiente virtual. Em seguida, são apresentadas duas técnicas para medir e monitorar desempenho de aplicações em máquinas virtuais e uma discussão sobre as métricas de desempenho a serem consideradas para avaliar o desempenho de aplicações executadas nesses ambientes.

4.1 Medição e Monitoramento

Em qualquer processo de medição de desempenho, o ponto chave é entender o que está sendo medido e o quão preciso e confiável os resultados numéricos são. Sendo assim, entender o funcionamento, a capacidade e as limitações das técnicas de medição de desempenho utilizadas para quantificar o desempenho de um sistema é uma questão essencial.

Para medir o nível de atividade de um sistema de computação utilizamos ferramentas de monitoramento [16], que têm como principal função monitorar e coletar informações sobre determinada operação realizada pelo sistema. Idealmente, um monitor deve ser um observador do sistema de computação estudado e não um participante do

sistema. O monitoramento deve ser feito de forma a não afetar a operação do sistema monitorado. Existem monitores de *hardware* e *software*. Os primeiros são ferramentas eletrônicas fisicamente acopladas ao sistema, de forma a detectar determinados eventos e capturar o estado de componentes de hardware, tais como registradores e canais de E/S. Os monitores de software consistem de rotinas inseridas no software do sistema com o intuito de armazenar determinados eventos e o estado do sistema.

4.2 Técnicas de Medição e Monitoramento

Um sistema operacional não consegue distinguir se está executando sobre uma máquina virtual ou sobre uma máquina real. Dessa forma, se alocarmos para uma máquina virtual apenas 30% da capacidade de um processador, quando o sistema operacional da máquina virtual reportar 100% de utilização, esses 100% na verdade correspondem a 30% da utilização do hardware verdadeiro adicionado ao processamento de operações de E/S e gerenciamento relativos a essa máquina virtual. Com base nesse exemplo, podemos notar que as técnicas convencionais utilizadas diretamente para medir desempenho e obter informações do sistema não podem ser utilizadas em ambientes virtuais. Esta seção discute duas técnicas para medição e monitoramento de aplicações executando sobre máquinas virtuais.

A forma utilizada para monitorar desempenho em um ambiente virtual pode variar de acordo com a precisão que se queira obter ou o nível de informação que se deseja extrair das máquinas virtuais e aplicações. Quando lidamos com um sistema operacional simples executando em uma máquina real, todas as informações do sistema são mantidas em um único local centralizado, o *kernel*. Em um ambiente virtual, dependendo das informações desejadas, as mesmas podem estar espalhadas pelas máquinas virtuais, sendo que muitas informações não podem ser acessadas diretamente pelo monitor de máquinas virtuais. Sendo assim, podemos definir duas maneiras de se monitorar um ambiente virtual: o monitoramento *centralizado* e o *distribuído*.

4.2.1 Monitoramento Centralizado

Métricas como utilização de CPU, acessos ao disco, consumo de memória e banda de rede consumida por cada máquina virtual podem ser obtidas no monitor de máquinas virtuais. Na camada do VMM encontra-se um escalonador de máquinas virtuais que determina quanto tempo cada máquina virtual ocupa a CPU. Conseqüentemente a porcentagem de tempo de CPU que cada máquina virtual ocupa durante um determinado período é uma informação que pode ser obtida no próprio monitor de máquinas virtuais. Da mesma forma, todos os acessos ao disco e envio ou recebimento de pacotes de rede de uma máquina virtual são contabilizados no IDD e armazenados na VMM. Além disso, informações sobre as características, configurações e estado das máquinas virtuais podem ser acessadas e monitoradas pelo VMM. Chamamos essa técnica de monitoramento centralizado, pois as informações sobre as máquinas virtuais estão centralizadas no monitor de máquinas virtuais. Como um exemplo, considere o cálculo da utilização de CPU. Podemos obter o tempo de CPU consumido por determinada máquina virtual em uma CPU e conseqüentemente criar ferramentas que reportam a utilização periodicamente. Uma ferramenta recentemente desenvolvida para monitorar a utilização de CPU de cada máquina virtual no ambiente virtual Xen é o *xentop*, incorporado pelo Xen 3.0 como um comando no VMM (*xm top*). Esta ferramenta funciona como o comando *top* do *Unix*, porém ela diferencia a quantidade de CPU utilizada entre todas as máquinas virtuais executadas simultaneamente. Esta técnica é extremamente prática e simples para medir a quantidade de recursos que cada máquina virtual está consumindo. Entretanto, ela não fornece nenhuma informação sobre as aplicações que executam nas máquinas virtuais ou mesmo qual aplicação é a maior responsável pela utilização de CPU de determinada máquina virtual. O monitor de máquinas virtuais oferece apenas uma visão do desempenho e do comportamento das máquinas virtuais como um todo.

4.2.2 Monitoramento Distribuído

Dependendo do nível de informação que se queira obter do sistema, centralizar o processo de coleta de informação no monitor de máquinas virtuais pode não ser possível. Como um exemplo, no ambiente virtual Xen, o monitor de máquinas virtuais não consegue determinar o processo em execução em uma máquina virtual, ou mesmo determinar se uma rotina de uma aplicação é a maior responsável por falhas nas caches. Nesse caso, precisamos monitorar contadores de hardware, aos quais apenas o VMM possui acesso, e obter o nome da função em execução, informação que está na máquina virtual. Uma forma para obtermos essas informações é introduzir monitores locais nas máquinas virtuais, de forma que o VMM se torna gerenciador desses monitores. A figura 4.1 ilustra este cenário. As máquinas virtuais são responsáveis pelo monitoramento e coleta de informações que o monitor de máquinas virtuais não pode acessar. As máquinas virtuais não possuem acesso ao hardware e precisam que o monitor de máquinas virtuais execute parte do processo. No caso do exemplo em que queremos saber qual aplicação é a maior responsável por falhas nas caches, o VMM passa a funcionar como uma interface de monitoramento para as máquinas virtuais, gerenciando seus acessos aos contadores de hardware. O monitor de máquinas virtuais pode programar contadores de hardware para disparar interrupções em intervalos regulares e pré-definidos de coleta de eventos. O VMM fica responsável por associar as interrupções geradas pelos contadores de hardware com suas respectivas máquinas virtuais. Ao receber esta opção as máquinas virtuais podem executar determinada ação como por exemplo, determinar a aplicação ou a função que gerou tal interrupção e registrar estas informações.

Ao utilizarmos essa técnica é necessário um cuidado com o número de interrupções causadas. Considere como exemplo, se monitorarmos o número de instruções executadas. Se lançarmos uma interrupção a cada instrução, o sistema claramente entraria em estado de *trashing*. Sendo assim, esta técnica faz sentido somente se coletarmos amostras, obtendo as informações desejadas em intervalos regulares da ocorrência do

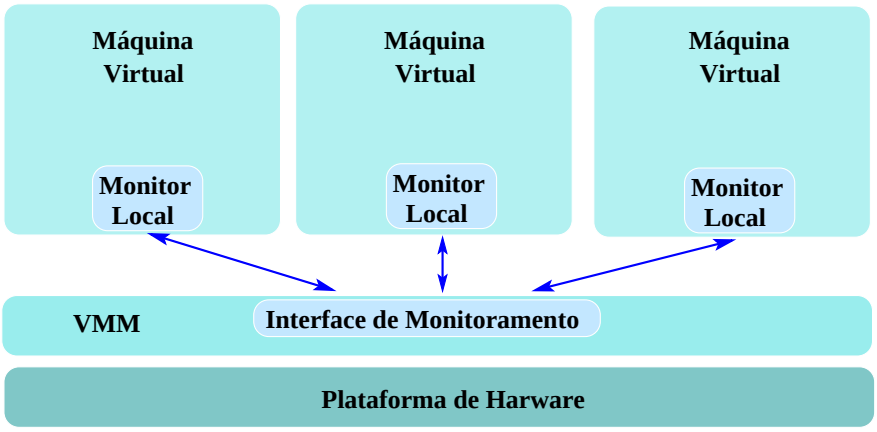


Figura 4.1: Arquitetura para medição de desempenho distribuída

evento monitorado. Por exemplo, a cada 100.000 falhas na cache L2 causadas por determinada máquina virtual, uma interrupção é lançada e uma amostra é coletada registrando a aplicação ou função em execução. Sendo assim, é necessário escolher um intervalo de amostragem adequado, para que o número de amostras seja significativo e o sistema não tenha seu desempenho afetado.

4.3 Métricas de Desempenho

Uma das questões que devemos levantar ao migrarmos aplicações de um servidor real para um servidor virtual é se as aplicações terão um desempenho equivalente em seu novo ambiente. Dessa forma, várias métricas utilizadas para se medir o desempenho de aplicações em servidores reais podem ser utilizadas para medir desempenho em ambientes virtuais. Como um exemplo, se considerarmos a migração de um servidor Web de uma plataforma real para um ambiente virtual poderíamos avaliar o desempenho do servidor Web em seu novo ambiente medindo: tempo de resposta, taxa de saída de requisições, número de requisições recusadas, etc. Essas métricas fornecem um entendimento do ponto de vista do usuário sobre o desempenho de aplicações em ambientes virtuais. Entretanto, para entendermos o comportamento de aplicações em ambientes virtuais é necessário analisar outras métricas. Esta seção mostra como estruturar a utilização de CPU de máquinas virtuais através das duas técnicas descritas anteriormente

e discute outras métricas úteis para o entendimento de ambientes virtuais.

4.3.1 Utilização de CPU

Em um ambiente virtual é necessário mais tempo de CPU do que em um ambiente real para se executar uma mesma aplicação. Sendo assim, a utilização de CPU é uma das métricas mais importantes para o estudo de desempenho de aplicações em ambientes virtuais. Um ambiente virtual possui várias camadas e cada elemento dessas camadas é responsável por parte da utilização dos recursos do hardware [27]. A seguir definimos a utilização de CPU nessas diferentes camadas de software.

Considere U^t_{cpu} como a utilização de CPU total de um sistema qualquer e $U_{cpu,r}$, a utilização de CPU pela classe r de instruções. Uma classe pode ser considerada como as operações realizadas por determinada camada do ambiente virtual, que são: instruções do VMM, instruções do sistema operacional (SO) em execução na máquina virtual e instruções dos programas executados sobre este SO.

Para formalizarmos a definição de utilização de CPU de uma máquina virtual vamos discutir a utilização de CPU de aplicações sendo executadas em três tipos de plataformas: (1) Hardware, (2) Sistema operacional e (3) Máquina virtual.

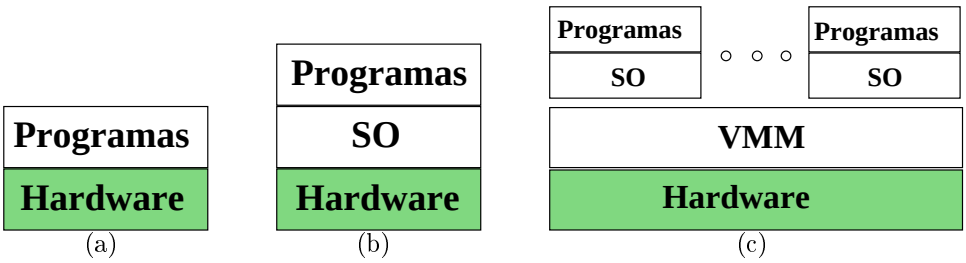


Figura 4.2: Diferentes ambientes de execução de aplicações

Hardware: Suponha uma aplicação que não utiliza o sistema operacional e acessa diretamente o hardware. Nesse caso, o programa tem controle completo da máquina, como mostra a figura 4.2(a). Observando a utilização de CPU sabemos que U^t_{cpu} re-

presenta a fração de tempo que a CPU ficou ocupada realizando apenas um tipo de atividade: executando instruções de programas. Sendo assim, podemos escrever que

$$U_{cpu}^{t} = U_{cpu,prog} \tag{4.1}$$

onde $U_{cpu,prog}$ se refere à fração de tempo de CPU consumida pelos programas em execução no hardware. Em outras palavras, uma única classe de instruções monopoliza o acesso ao hardware.

Sistema Operacional: O próximo ambiente consiste de um sistema operacional gerenciando os recursos de hardware. Aplicações, por sua vez, são executadas sobre o sistema operacional, como ilustrado na figura 4.2(b). A utilização de CPU neste caso indica a fração de tempo que a CPU está ocupada fazendo dois tipos de atividades: processando instruções de programas e executando rotinas do sistema operacional. Os recursos de hardware gastos com o sistema operacional são conhecidos como *overhead*. Dessa forma, a utilização de CPU total do sistema pode ser escrita da seguinte forma:

$$U_{cpu}^{t} = U_{cpu,so} + U_{cpu,prog} \tag{4.2}$$

onde $U_{cpu,so}$ corresponde ao *overhead* do sistema operacional, caracterizado por atividades, tais como capturar operações de E/S, paginação e memória virtual.

Máquina Virtual: Em um ambiente de execução fornecido por um sistema virtual a CPU é compartilhada por diferentes camadas de *software* como mostra a figura 4.2(c). O monitor de máquinas virtuais realiza diversas operações relativas ao acesso ao hardware e gerenciamento das máquinas virtuais, como por exemplo operações de E/S. Seja Dom_1, Dom_2, ..., Dom_N máquinas virtuais compartilhando a mesma estrutura física. A utilização de CPU de uma única máquina virtual, $U_{cpu}^{Dom_i}$ pode ser expressa por:

$$U_{cpu}^{Dom_i} = U_{cpu,vmm}^{Dom_i} + U_{cpu,so}^{Dom_i} + U_{cpu,prog}^{Dom_i} \tag{4.3}$$

onde $U_{cpu,vmm}^{Dom_i}$ representa a utilização de CPU consumida pelo monitor de máquinas virtuais para a gerenciar as operações de E/S e traduzir as instruções da máquina virtual Dom_i, $U_{cpu,so}^{Dom_i}$ corresponde à utilização de CPU devido ao *overhead* do sistema operacional e $U_{cpu,prog}^{Dom_i}$ é a utilização de CPU para rodar os programas executados na máquina virtual Dom_i. De uma maneira mais geral, a utilização de CPU do sistema pode ser escrita como a soma da utilização de CPU consumida por todas as máquinas virtuais e seus respectivos sistemas operacionais e aplicações:

$$U_{cpu}^{t} = \sum_{i=1}^{N} U_{cpu}^{Dom_i} \tag{4.4}$$

onde N é o número de máquinas virtuais em execução no VMM.

Dependendo do tipo de análise e monitoramento realizada no sistema, podemos verificar qual a porcentagem de CPU gasta em cada uma dessas camadas. No caso em que migramos um determinado serviço de um ambiente real para um ambiente virtual é importante saber qual o *overhead* introduzido pela virtualização, ou seja, estamos interessados em saber $U_{cpu}^{Dom_i}$ para compararmos com a utilização das mesmas aplicações executando em uma plataforma não virtual. A próxima seção mostra como calcular $U_{cpu}^{Dom_i}$ utilizando as técnicas descritas de medição de desempenho descritas anteriormente.

4.3.1.1 Calculando Utilização de CPU em um Ambiente Virtual

No capítulo 3 discutimos que o IDD é responsável por todas as operações de E/S das máquinas virtuais. Dessa forma, para calcularmos $U_{cpu}^{Dom_i}$, a utilização de CPU de uma determinada máquina virtual, precisamos levar em consideração a utilização de CPU do IDD para realizar as operações de E/S da máquina virtual Dom_i. Vamos

designar o termo Dom_0 para representar IDD [1].

Para calcularmos a utilização de CPU a partir da técnica centralizada no VMM, podemos realizar duas medições, $M_0^{Dom_i}$ e $M_t^{Dom_i}$, do tempo de CPU consumido pela máquina virtual Dom_i em um intervalo de tempo T. Sendo assim, definimos B^{Dom_i} como o tempo em que a CPU ficou ocupada com a máquina virtual Dom_i no intervalo de tempo T e $B_{Dom_0}^{Dom_i}$ como o tempo em que a CPU ficou ocupada por Dom_0 para realizar operações de E/S da máquina virtual Dom_i. B^{Dom_i} pode ser calculado da seguinte forma:

$$B^{Dom_i} = M_t^{Dom_i} - M_0^{Dom_i} \tag{4.5}$$

Note que B^{Dom_i} não inclui $B_{Dom_0}^{Dom_i}$. No caso em que temos apenas uma máquina virtual, Dom_1, executando no sistema e Dom_0 não realiza nenhuma outra operação a não ser as operações de E/S da máquina virtual em execução podemos calcular $B_{Dom_0}^{Dom_1}$ como na equação 4.5. Entretanto, quando existe mais de uma máquina virtual sendo executada simultaneamente, podemos estimar $B_{Dom_0}^{Dom_i}$ a partir do número de trocas de página entre o dom_0 e dom_i [11]. Como discutido no capítulo 3, o modelo de E/S do ambiente virtual Xen (também adotado por outros sistemas como o Denali) funciona da seguinte forma. Para evitar o custo de copiar um pacote de dados de E/S da máquina virtual para o dom_0, o Xen implementa um mecanismo de troca de páginas na memória. Sendo assim, o custo para o Dom_0 processar pacotes de 36 bytes e pacotes de 1448 bytes seria o mesmo, caso o número de troca de páginas na memória para entregar estes pacotes seja o mesmo [11]. Sendo assim, para calcular $B_{Dom_0}^{Dom_i}$ em um ambiente virtual com mais de uma máquina virtual em execução precisamos primeiramente calcular, $B_{Dom_0}^{tp}$, o tempo que uma operação de troca de páginas na memória ocupa a CPU do Dom_0. Se medirmos o número total de trocas de páginas $N_{Dom_0}^{t}$ ocorridas no Dom_0, causadas por todas a máquinas virtuais em execução e calcularmos B^{Dom_0}, o tempo

[1] Na terminologia do Xen o IDD é, às vezes, chamado de *Domain 0*, pelo fato do IDD ser a máquina virtual 0

que em que a CPU ficou ocupada com Dom_0, através da equação 4.5, temos:

$$B^{tp}_{Dom_0} = \frac{B^{Dom_0}}{N^t_{Dom_0}} \tag{4.6}$$

Dessa forma, medindo o número de troca de páginas em uma máquina virtual, N_{Dom_i}, temos:

$$B^{Dom_i}_{Dom_0} = N_{Dom_i} B^{tp}_{Dom_0} \tag{4.7}$$

Considerando-se apenas uma CPU para o Dom_i, podemos definir $U^{Dom_i}_{cpu}$ como:

$$U^{Dom_i}_{cpu} = \frac{B^{Dom_i} + B^{Dom_i}_{Dom_0}}{T} \tag{4.8}$$

Observe que a utilização de CPU de uma máquina virtual calculada com o método acima consiste de uma média da utilização de CPU no período analisado. Como exemplo, considere uma máquina virtual Dom_1 sendo monitorada em um intervalo de 60 segundos. Se observarmos que a CPU ficou ocupada 30 segundos com esta máquina virtual e 18 segundos com dom_0 para realizar operações de E/S para esta máquina virtual temos a utilização média de CPU neste intervalo: $U^{Dom_1}_{cpu} = \frac{30+18}{60} = 0,8$.

Se utilizarmos a técnica distribuída podemos obter a utilização de CPU de determinada máquina virtual através de amostras de eventos de hardware. Nesse caso, o evento de hardware monitorado pode ser o número de ciclos de clock em que a CPU ficou ocupada.

O cálculo da utilização de CPU para cada máquina virtual através desta técnica pode ser realizado da seguinte forma. Seja s_0, s_1,...,s_N o número de amostras coletadas para cada uma das N máquinas virtuais $Dom_0, Dom_1, ..., Dom_N$ durante o intervalo de tempo T do experimento. Para cada máquina virtual, uma amostra é coletada a cada intervalo de C ciclos de *clock* ocupados e o processador da máquina que hospeda as máquinas virtuais possui freqüência F Hz. Podemos estimar B^{Dom_i} da seguinte forma:

$$B^{Dom_i} = \frac{s_i\ C}{F} \tag{4.9}$$

Conseqüentemente, a utilização de CPU da máquina virtual Dom_i pode ser calculada utilizando-se a equação 4.8. Note que, novamente $B^{Dom_i}_{Dom_0}$ só pode ser calculado através da equação 4.9 se tivermos apenas uma máquina virtual no sistema e o Dom_0 não realizar nenhuma outra operação a não ser as operações de E/S da máquina virtual em execução.

Como um exemplo, considere um processador de $2x10^9$ Hz executando um ambiente virtual com apenas uma máquina virtual Dom_1. Uma amostra é coletada a cada intervalo de 10^6 ciclos de clock ocupados. Suponha que durante um período de monitoramento do sistema de 100 segundos foram coletados 1000 amostras para a máquina virtual Dom_1 e 500 amostras para Dom_0. Sendo assim, temos: $B^{Dom_1} = \frac{1000x10^6}{2x10^9} = 50 segundos$ e $B^{Dom_1}_{Dom_0} = \frac{500x10^6}{2x10^9} = 25\ segundos$. Conseqüentemente $U^{Dom_1}_{cpu} = \frac{50+25}{100} = 0,75$.

Observe que a utilização de CPU estimada com esta métrica pode não ser precisa se o intervalo C de coletas de amostras for grande relativo ao número de eventos coletados. Entretanto, se este intervalo for pequeno, o desempenho do sistema pode ficar comprometido devido ao grande número de interrupções causadas pela coleta de amostras. A grande vantagem desta técnica é permitir que outras informações sejam coletadas juntamente com as amostras. Podemos, por exemplo, estar interessados em saber qual o conjunto de rotinas do TCP é responsável pela maior parte da utilização de CPU quando executamos uma aplicação que realiza um grande número de conexões TCP.

4.3.2 Outras Métricas de Desempenho em Ambientes Virtuais

Como destacamos anteriormente, boa parte das métricas convencionais para avaliação de desempenho de aplicações em ambientes reais podem ser utilizadas para o entendimento do desempenho de aplicações no ambiente virtual. Entretanto, quando

utilizamos a técnica de monitoramento distribuído, vários eventos de hardware podem se tornar métricas de desempenho de um sistema virtual. Como um exemplo, quando colocamos mais de uma máquina virtual compartilhando o mesmo hardware, é importante verificar se a causa da queda no desempenho do ambiente virtual não é causada por um grande número de falhas na cache. Em um servidor multi-processado pode ser mais eficiente agrupar em um mesmo processador aplicações que não utilizam tanto caches e deixar aplicações que carregam grandes porções da memória rodarem em uma CPU exclusiva.

Capítulo 5

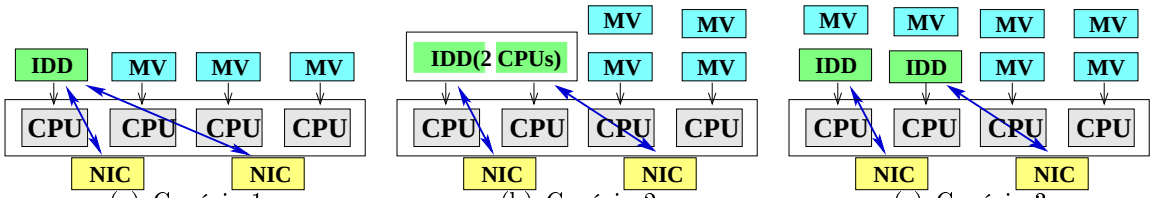
Estudo de Caso

Este capítulo apresenta um estudo de caso que utiliza as técnicas de medição de desempenho discutidas no capítulo 4 e explora a questão da migração de aplicações de servidores reais para servidores virtuais. Inicialmente enunciamos o problema abordado através de um exemplo e em seguida descrevemos os experimentos realizados, ambiente experimental, conjunto de testes, ferramentas e métricas utilizadas. Os resultados do estudo de caso são apresentados no próximo capítulo.

5.1 Configurando um Servidor Virtual

Para ilustrarmos o problema abordado neste estudo de caso vamos apresentar um exemplo motivador que levanta várias questões relativas à execução de aplicações em servidores virtuais. Considere uma máquina com quatro processadores e duas interfaces de rede (NIC - *Network Interface Card*), na qual queremos implantar um ambiente virtual com várias aplicações executando ao mesmo tempo. A figura 5.1 mostra três cenários, cada um representando uma configuração diferente para os mesmos recursos. A figura 5.1(a) ilustra um cenário onde ambos os NICs são associados ao mesmo IDD, executando em uma única CPU. A questão principal que podemos levantar neste cenário está ligada à escalabilidade do IDD. Uma única CPU para o IDD seria suficiente para suportar o processamento do tráfego de E/S das máquinas virtuais? Outra

questão importante consiste em definir um conjunto de características das aplicações que permite alocar recursos para o IDD de forma a garantir o desempenho esperado das máquinas virtuais. Na figura 5.1(b) temos um IDD executando com SMP (*Symmetric Multi-Processing*) em dois processadores e, para cada processador, associamos um dos NICs. Observando este cenário podemos levantar as seguintes perguntas: Esta configuração pode melhorar a escalabilidade do IDD? Quanto? Como terceiro cenário, representado na figura 5.1(c) apresentamos dois IDDs, cada um executando em uma CPU. Nesta configuração, cada NIC é associado a um IDD[1]. Comparando este cenário com a segunda configuração, seria interessante saber qual configuração leva a um melhor desempenho. Outro importante assunto a ser estudado está relacionado ao número de máquinas virtuais em cada CPU e à interferência, em termos de desempenho, que máquinas virtuais podem causar umas nas outras quando compartilham os mesmos recursos.



(a) Cenário 1 (b) Cenário 2 (c) Cenário 3

Figura 5.1: Diferentes configurações para os mesmos recursos

Neste estudo de caso, são abordadas várias das questões levantadas nesse exemplo, utilizando um servidor similar ao descrito na figura 5.1. Entender as razões do overhead da virtualização e quantificar este overhead é crucial para implantarmos e configurarmos um ambiente virtual. O foco da avaliação de desempenho apresentada está em estudar experimentalmente o desempenho do ambiente virtual Xen em um servidor como uma função das características das aplicações e escolhas de configuração. Uma outra alternativa seria modelagem analítica. Mas como ainda não há dados suficientes para a construção de um modelo, deixamos esta alternativa como trabalho futuro.

[1]A versão atual do Xen (3.0 em desenvolvimento) ainda não possui suporte para múltiplos IDDs

5.2 Metodologia Utilizada

Neste estudo de caso, criamos um cenário muito semelhante ao do exemplo anterior, no qual queremos implantar aplicações executando em um servidor virtual. O servidor utilizado possui quatro processadores e duas interfaces de rede e o ambiente virtual utilizado é o Xen. Cada experimento consiste na execução de um determinado conjunto de testes em um ambiente real e virtual, que são monitorados por ferramentas de monitoramento de desempenho que utilizam as técnicas discutidas no capítulo 4. O restante desta seção descreve o ambiente experimental, os conjuntos de testes e as ferramentas utilizadas.

5.2.1 Ambiente experimental

Para a avaliação experimental realizada neste trabalho foram utilizados os seguintes recursos. Para todos os experimentos utilizamos uma máquina Compaq Proliant DL360 com quatro CPUs *Intel Xeon server* de 2800 MHz com 4 GB de memória RAM e duas placas de rede Broadcom 'Tigon 3' Gigabit. Cada NIC conecta o servidor a duas máquinas clientes através de um *switch* Gigabit. As máquinas clientes rodam Linux com Kernel 2.6.12.

Para todos os experimentos utilizamos a versão 3.0 do Xen, ainda em desenvolvimento. As máquinas virtuais executam XenoLinux derivado a partir do *Kernel* 2.6.12 do Linux. Todas as máquinas virtuais são configuradas com 256 MB de memória RAM. Para todos os experimentos o BVT (*Borrowed Virtual Time*) foi utilizado como escalonador do Xen. Para compararmos o desempenho do ambiente virtual com um ambiente real, utilizamos o sistema operacional Linux com *Kernel* 2.6.12 sobre a mesma plataforma de hardware do ambiente virtual. Como ferramentas de medição de desempenho, utilizamos o Xencpu descrito na seção 5.2.3.1. Além disso, utilizamos o Xenoprof e o Oprofile [33] para contar eventos de hardware e medir utilização de CPU para o Xen o Linux, respectivamente. A seção 5.2.3.2 descreve o Xenoprof e seu uso nos experimentos. Foi utilizado o Oprofile versão 0.8.2 em e o Xenoprof foi derivado a partir desta

mesma versão do Oprofile.

5.2.2 Conjuntos de Testes

A seguir descrevemos três conjuntos de testes utilizados nos experimentos:

- **Emissão e recepção de tráfego TCP**: Foi utilizado um conjunto de testes de rede similar ao TTCP [15], que mede taxa de saída e de chegada ao enviar ou receber tráfego TCP para um pequeno número de conexões. A idéia principal desse conjunto de testes é medir a utilização de CPU em ambientes virtuais e não virtuais para diferentes taxas de processamento de rede, taxa de pacotes e tamanho dos pacotes. Foi desenvolvida uma pequena aplicação emissora e outra receptora de tráfego TCP. O processo emissor funciona em um laço contínuo enviando pacotes de determinado tamanho, passado como parâmetro. Para obtermos diferentes taxas de saída e taxa de chegadas de pacotes, utilizamos um mecanismo de *desaceleração* para reduzir a taxa em que os pacotes são enviados. O mecanismo consiste em executar no lado da aplicação do emissor um laço que não executa nenhum comando logo após cada pacote enviado. O tamanho deste laço determina a taxa de saída adquirida. Para avaliar o lado do servidor, utilizamos um mecanismo semelhante do lado do receptor. Para garantir o envio de pacotes de mesmo tamanho, segmentação TCP foi desabilitada modificando o tamanho máximo do segmento e desabilitando-se o algoritmo de Nagle [30].

- **Servidor Web**: Outro conjunto de testes utilizado consiste em gerar carga para um servidor Web que recebe requisições geradas a partir de uma carga sintética. Este conjunto de testes estressa todo o sistema de rede, incluindo os caminhos de transferência de dados e estabelecimento e término de conexão para um grande número de conexões. Utilizamos o *httperf* [31] como clientes e o Apache [5] versão 2.0.54 como servidor Web. O *httperf* é uma ferramenta que permite gerar várias cargas HTTP e medir o desempenho do servidor do ponto de vista do cliente. Para

medir a taxa de processamento de rede do servidor Web, invocamos o *httperf* em uma máquina cliente, que envia requisições ao servidor a uma taxa fixa e mede a taxa na qual as respostas chegam. A tabela 5.2.2 mostra algumas características do conteúdo disponibilizado no Servidor Web. As requisições realizadas pelo *httperf* fazem parte do conjunto de cargas do SPECWeb99 [32] e não possuem conteúdo dinâmico.

- **Conjunto de testes de CPU**: Para estudarmos a interferência que uma máquina virtual pode causar em outras quando elas compartilham o mesmo processador, utilizamos uma aplicação que faz uso intenso de CPU. A aplicação utilizada consiste da compilação do código do Kernel 2.6.11 do Linux. Esta aplicação é executada por um intervalo significativo de tempo e executa uma quantidade representativa de funções capazes de gerar um grande número de falhas nas caches.

Número de arquivos distintos	2196
Tamanho total carga	297 MB
Menor arquivo	0 KB
Maior arquivo	900 KB
Média do tamanho	138,87 KB
Coeficiente de variabilidade	1,77
Primeiro quartil do tamanho	1 KB
Mediana do tamanho	20 KB
Terceiro quartil do tamanho	100 KB

Tabela 5.1: Características da Carga Web

O conjunto de testes de rede e o servidor Web testam diferentes aspectos da pilha de rede e a aplicação escolhida como conjunto de testes de CPU representa bem uma grande aplicação real que utiliza 100% da CPU executando um grande número de funções diferentes. Os detalhes sobre a utilização dos conjuntos de testes nos experimentos estão descritos no capítulo 6.

5.2.3 Ferramentas Utilizadas

Para medir utilização de CPU e obter outras informações relativas ao desempenho foram utilizadas duas ferramentas de monitoramento. A primeira delas, chamada Xencpu, utiliza a técnica centralizada para capturar a utilização de CPU das máquinas virtuais. A segunda ferramenta é o Xenoprof que utiliza a técnica de medição de desempenho distribuída para obter informações sobre as máquinas virtuais.

5.2.3.1 Xencpu

O Xencpu é uma ferramenta que desenvolvemos baseada no código de outra ferramenta chamada Xentop. O *Xentop* funciona como o comando *top* do unix, porém ele diferencia a quantidade de CPU utilizada entre todas as máquinas virtuais executadas simultaneamente. O Xentop foi recentemente incluído no código do Xen e pode ser acessado através do comando *xm top*. A ferramenta Xencpu utiliza os mesmos princípios do Xentop, porém recebe como parâmetro a CPU virtual e imprime o tempo em que a CPU ficou ocupada com determinada máquina virtual até aquele momento. Dessa forma, utilizamos esta ferramenta no início e no final de cada experimento para calcular a utilização de CPU de acordo com a equação 4.8.

Esta ferramenta fornece menos informações do que o Xenoprof. Entretanto, para experimentos em que queremos obter somente a utilização de CPU, o Xencpu é mais simples e fácil de usar. Esta ferramenta foi desenvolvida porque o Xenoprof não funciona com SMP. Para os resultados da seção 6.3, realizamos experimentos com duas CPUs no IDD e desenvolvemos esta ferramenta para medir utilização de CPU, uma vez que o Xenoprof não funciona com aquela configuração. Em todos os outros experimentos utilizamos o Xenoprof, discutido na próxima seção.

5.2.3.2 Xenoprof

O Xenoprof é uma ferramenta que coleta estatísticas de eventos de hardware de ambientes virtuais do Xen. Esta ferramenta foi proposta em [29]. Esse trabalho descreve

a arquitetura do Xenoprof e utiliza o Xenoprof para detectar partes do código do Xen que podem ser melhoradas, porém, uma avaliação de desempenho da ferramenta não é apresentada. Como neste trabalho utilizamos o Xenoprof para calcular métricas de desempenho, como utilização de CPU, é essencial verificarmos o impacto do Xenoprof no desempenho do sistema, além de definir um período ideal para coleta de amostras. O código do Xenoprof é aberto, livre e está disponível na Internet [49].

Amostras Coletadas	**Porcentagem do Total**	**Função**
10602	56,2799	vmlinux-syms-2.6.12-xenU
2846	15,1078	xen-syms-3.0-devel
2126	11,2857	httpd
1321	7,0124	libapr-0.so.0.9.6
942	5,0005	libc-2.3.2.so
632	3,3549	libaprutil-0.so.0.9.6
316	1,6775	libpthread-0.10.so
21	0,1115	ld-2.3.2.so
17	0,0902	libcrypto.so.0.9.7a
10	0,0531	oprofile
2	0,0106	bash
2	0,0106	oprofiled
1	0,0053	init

Tabela 5.2: Exemplo de saída do Xenoprof

O Xenoprof utiliza monitoramento de eventos de hardware para coletar amostras periódicas de dados de desempenho. O código do Xenoprof é baseado em uma ferramenta do Linux chamada Oprofile [33], que tem a mesma funcionalidade do Xenoprof para Linux. O Xenoprof é capaz de determinar a distribuição dos eventos entre rotinas de cada uma das máquinas virtuais executando no sistema. Considere, por exemplo, que estejamos executando um servidor Web em uma máquina virtual hóspede e utilizando o Xenoprof para coletar amostras do número de ciclos em que a CPU ficou ocupada. A tabela 5.2 mostra um exemplo da saída do Xenoprof. A terceira coluna da tabela mostra a função em execução no momento da coleta da amostra. A primeira coluna mostra o número total de amostras coletadas para cada função enquanto que a segunda coluna mostra a porcentagem que este número representa em relação ao total.

O Xenoprof pode ser utilizado para estudar os custos de desempenho da virtuali-

zação e as interações entre múltiplas máquinas virtuais. Além de falhas nas caches, utilizamos o Xenoprof para medir utilização de CPU, que permite identificar quais rotinas estão consumindo mais ciclos de CPU.

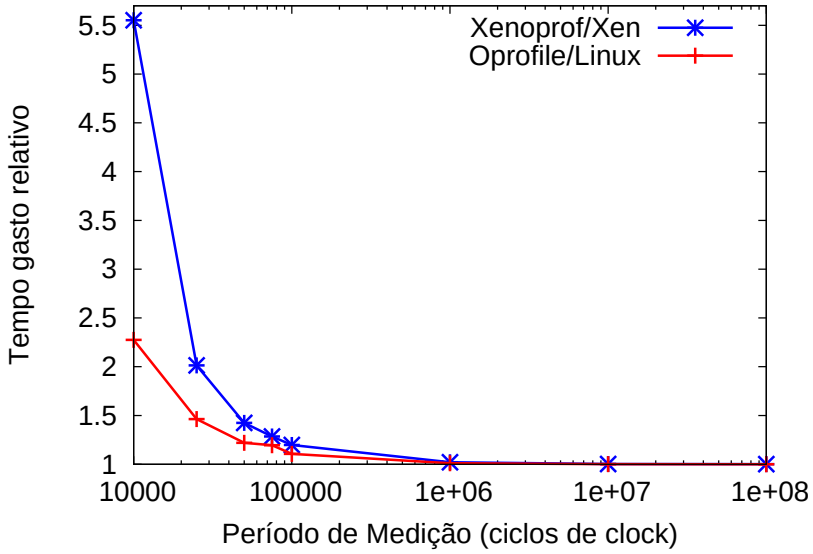


Figura 5.2: Desempenho do Xenoprof e do Oprofile

Para comparar as medições do Xenoprof no Xen com as medições do Oprofile no Linux, é importante definir um intervalo para coletar as amostras que permita uma comparação justa entre os dois sistemas. Para isso, executamos o conjunto de testes de CPU em ambos os sistemas e medimos o tempo de execução para diferentes intervalos de coleta de eventos de hardware. O evento utilizado foi o número de ciclos de clock ocupados. A figura 5.2 compara o tempo de execução do conjunto de testes para uma máquina utilizando Linux executando Oprofile e a mesma máquina utilizando Xen e executando Xenoprof. Os valores apresentados correspondem ao tempo relativo para executar a aplicação rodando o Xenoprof ou Oprofile comparado com Xen e Linux sem rodar estas ferramentas. Para valores pequenos de períodos de amostragem tanto o Xen quando o Linux começam a entrar em *trashing* devido ao alto número de interrupções causadas pelas ferramentas. Esse efeito do *trashing* é mais perceptível no Xen, pois existe mais código sendo executado neste sistema devido à virtualização. Entretanto, para um período maior de coleta de estatísticas, nenhuma das ferramentas interfere

significativamente no desempenho do sistema. Para períodos de 10^6 eventos, o Xenoprof causa 2% a mais de degradação do desempenho do sistema comparado com o Oprofile e, para 10^7 ciclos ocupados, o Xenoprof causa apenas 0,25% a mais de degradação comparado com o Oprofile. Com o intuito de garantir uma comparação justa entre os dois sistemas, escolhemos 10^7 ciclos de *clock* ocupados como período utilizado em todos os experimentos.

É importante ressaltar que, para cada evento monitorado, devemos escolher um período de coleta de amostras adequado, pois diferentes eventos ocorrem com freqüências diferentes. No caso de falhas em cache escolhemos um intervalo de 10^4, que é suficiente para não interferir no desempenho do sistema.

Capítulo 6

Resultados

Este capítulo apresenta uma avaliação de desempenho do modelo de E/S atual do Xen como uma função de configurações do ambiente virtual e características do tráfego das aplicações. Para cada avaliação o ambiente virtual do Xen é comparado com um ambiente Linux não virtual, ambos executados na mesma plataforma de hardware. As análises apresentadas cobrem vários problemas de desempenho que surgem na implantação de aplicações em servidores virtuais, bem como explora configurações e características das aplicações que podem ser úteis para alocação de recursos em um ambiente virtual. Inicialmente apresentamos uma visão geral do desempenho do processamento de E/S de rede do ambiente virtual Xen comparado com o sistema Linux. Em seguida avaliamos a utilização de CPU para processamento de operações de E/S no IDD. Na tentativa de melhorar a escalabilidade do IDD, foram exploradas diferentes configurações para o IDD e, finalmente, estudamos a interferência entre máquinas virtuais que compartilham os mesmos recursos físicos.

6.1 Processamento de E/S de rede no Xen

Inicialmente comparamos o desempenho de rede de uma aplicação em uma máquina virtual do Xen com a mesma aplicação executando no Linux, que utilizamos como linha de base para comparação de desempenho. O ambiente virtual é configurado com o IDD

e a máquina virtual executando em CPUs diferentes. Foi utilizado neste experimento o conjunto de testes que recebe tráfego TCP e a máquina virtual receptora está conectada a uma máquina cliente Linux, que envia tráfego por uma interface de 1 Gigabit/s.

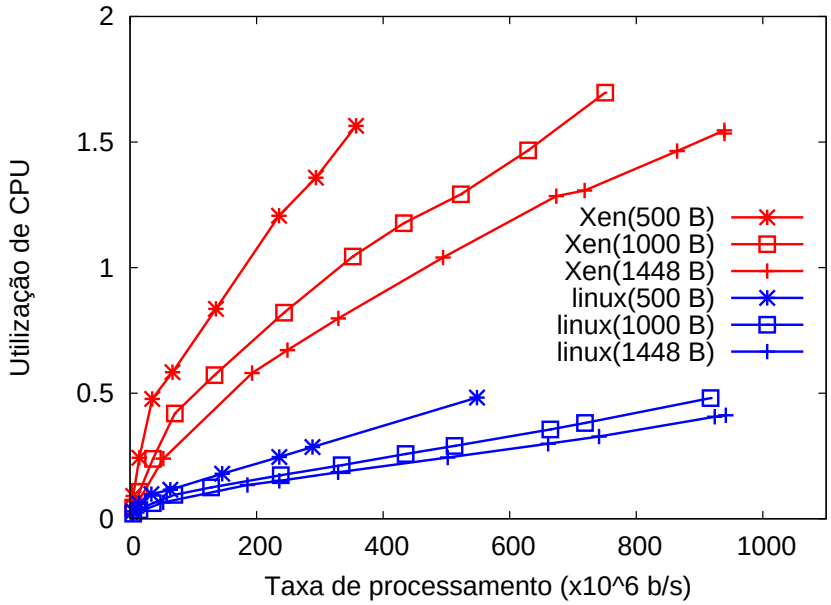


Figura 6.1: Utilização de CPU para Linux e Xen para diferentes taxas de processamento e tamanhos de pacotes

A figura 6.1 compara o desempenho do conjunto de testes que recebe tráfego TCP na máquina virtual e no Linux. Este gráfico mostra a soma dos dois processadores utilizados como uma função da taxa de processamento de rede para diferentes tamanhos de pacotes para Linux e Xen. O tamanho dos pacotes está representado entre parênteses na figura. Note que, para todos os tamanhos de pacotes avaliados, a utilização de CPU do Xen é consideravelmente maior do que no Linux. Em outras palavras, a alocação de CPU necessária para suportar o processamento de E/S de rede no Xen é cerca de 3 a 5 vezes maior do que no sistema Linux. Note que quando comparamos a utilização de CPU consumida pelo processamento do tráfego de pacotes de tamanhos diferentes, obtemos um maior consumo de tempo de CPU para o tráfego de pacotes menores para uma mesma taxa de processamento. Este processamento é causado porque é preciso enviar um maior número de pacotes pequenos para se atingir a mesma taxa de serviço dos pacotes maiores.

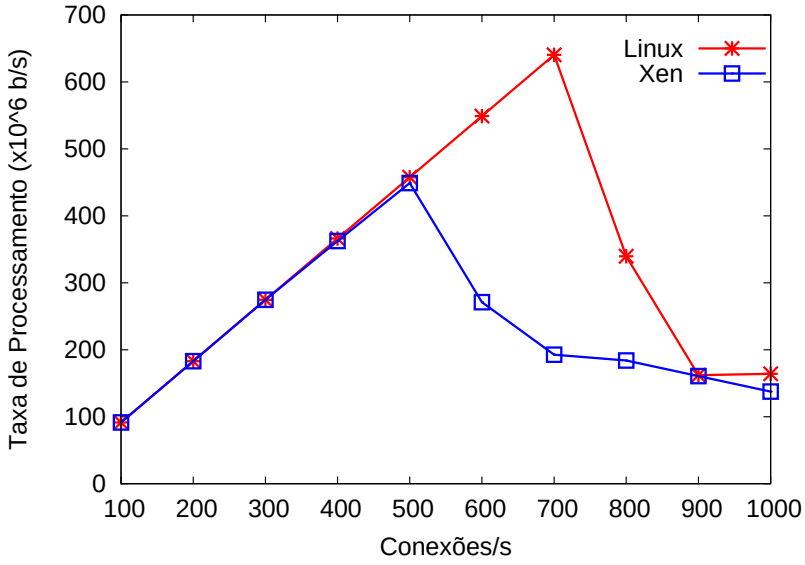


Figura 6.2: Taxa de processamento de rede para Linux e Xen à medida que aumentamos o número de conexões/s

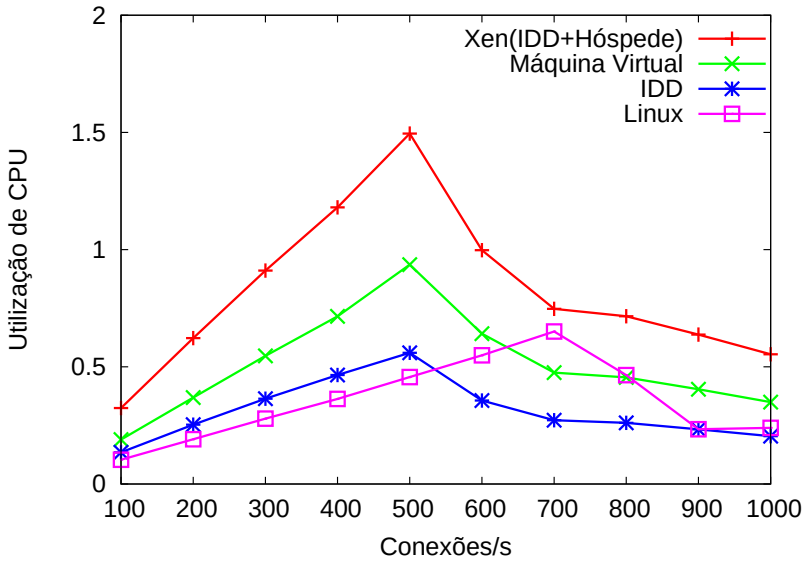


Figura 6.3: Utilização de CPU para Linux e Xen à medida que aumentamos o número de conexões/s

Em seguida avaliamos o desempenho de rede do Xen utilizando o conjunto de testes do servidor Web. As figuras 6.2 e 6.3 comparam o desempenho do servidor Web em uma máquina virtual com o desempenho do servidor rodando na mesma plataforma com Linux. Como podemos notar na figura 6.2, as taxas de serviço observadas para o sistema Linux e o ambiente virtual são praticamente as mesmas à medida que aumentamos a

carga no servidor Web até 500 conexões/s. Após esse ponto o desempenho da máquina virtual começa a cair. Este efeito pode ser entendido ao analisarmos a figura 6.3. Esta figura mostra a utilização de CPU do sistema Linux, da máquina virtual hóspede, do IDD e também a soma das utilizações da máquina virtual e do IDD como uma função do número de conexões por segundo. No ponto onde o desempenho do servidor Web começa a degradar no ambiente virtual, a CPU da máquina virtual atinge seu máximo de utilização, saturando o sistema. Investigando o motivo da degradação no desempenho, descobrimos que após este ponto a fila de SYN do TCP estourou para valores muito altos de números de conexões/s, o que causou um grande número de conexões recusadas e conseqüentemente diminuiu a utilização da CPU. Como menos pacotes são processados a utilização de CPU e a taxa de processamento diminuem. Este mesmo efeito foi observado no Linux, entretanto, o limitador do sistema neste caso foi o número máximo de conexões aceitas pelo servidor Web.

Note que as taxas de processamento de rede do servidor Web na máquina virtual e no Linux são muito próximas até 500 conexões/s e, há um custo de 3,3 vezes mais utilização de CPU para o ambiente virtual. Este resultado reforça os resultados para o conjunto de testes receptor de tráfego TCP e contrasta com resultados de pesquisas anteriores [9], que afirmam que o Xen obtém um desempenho próximo do Linux. Naqueles experimentos, o NIC era o ponto de contenção do sistema. Como o sistema não era limitado pela CPU, uma maior utilização de CPU com o Xen não foi refletida na redução da taxa de processamento. Esta mesma observação foi feita em [29].

Sumarizando os resultados desta seção, o Xen consome de 3 a 5 vezes mais utilização de CPU para prover uma taxa de processamento de rede próxima à do sistema Linux.

6.2 Entendendo o Processamento de E/S no IDD

Nesta seção, analisamos as principais causas do processamento de E/S no IDD a fim de entendermos os fatores que influenciam este processamento.

Para estas análises foi utilizado o conjunto de testes receptor de tráfego TCP. O

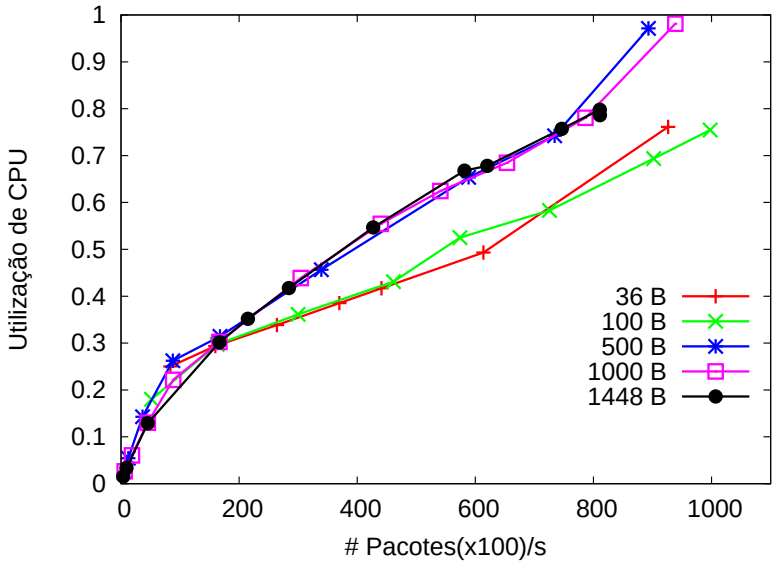


Figura 6.4: Utilização de CPU no IDD x pacotes de dados/s

ambiente virtual foi configurado com o IDD em uma CPU e a máquina virtual recebendo pacotes de dados em outra CPU. A figura 6.4 mostra a utilização de CPU no IDD como uma função do número de pacotes por segundo para tamanhos diferentes de pacotes. É esperado que a utilização de CPU do IDD para processamento de E/S seja a mesma para pacotes de tamanhos diferentes. Como descrito anteriormente na seção 3.2, quando o IDD intermedia o tráfego de rede de uma máquina virtual, o Xen utiliza um mecanismo de compartilhamento de memória em que o Xen mapeia a página com o dado na máquina virtual em troca de uma página não utilizada da máquina virtual, o que não depende do tamanho do pacote. Entretanto, não é isso que observamos na figura 6.4. Podemos notar dois comportamentos distintos no gráfico, um para pacotes menores e outro para os pacotes maiores. Com o intuito de entender este comportamento, investigamos o número de pacotes de *acks* enviados pela máquina virtual. A figura 6.5 mostra o número de *acks* por pacotes de dados como uma função do número de pacotes de dados por segundo. Note que há dois comportamentos distintos para a proporção de *acks* por pacotes de dados enviados, o que claramente interfere na utilização de CPU do IDD. Conseqüentemente, a utilização de CPU do IDD realmente não é sensível ao tamanho dos pacotes. O desempenho do IDD é sensível ao número

total de pacotes, recebidos e enviados pelas máquinas virtuais. Nos resultados das figuras 6.4 e 6.5, o total de pacotes corresponde ao número de pacotes de dados somado ao número de *acks*. Para uma descrição detalhada do comportamento dos pacotes de *acks* no TCP vide a RFC3390 [3].

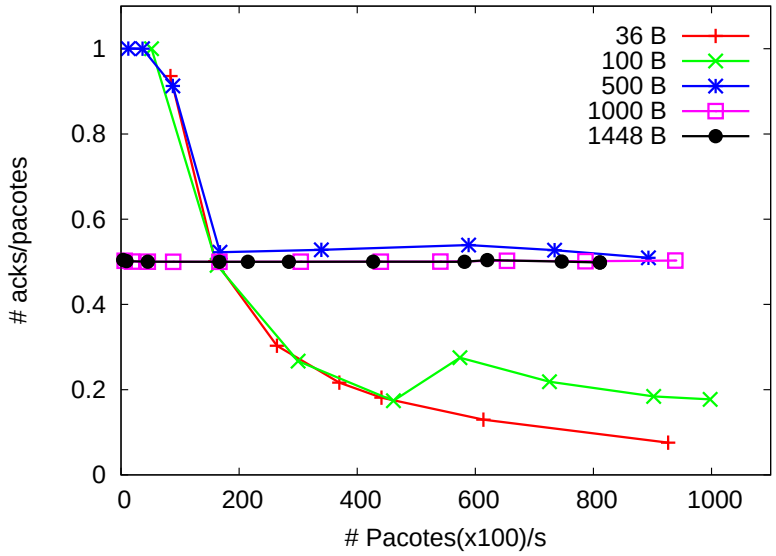


Figura 6.5: Acks/pacotes de dados x pacotes de dados/s

Outra importante observação sobre a figura 6.4 é que o IDD é o ponto de contenção do sistema para pacotes de 500 e 1000 bytes. Apesar de que aplicações reais devem consumir mais processamento de CPU no lado da máquina virtual comparado com o conjunto de testes utilizado, é evidente que o IDD não escala bem, e pode se tornar o limitador de tráfego de E/S para poucas máquinas virtuais. Na próxima seção avaliamos o desempenho de diferentes configurações do ambiente virtual na tentativa de melhorar a capacidade do IDD para processamento de E/S.

Sumarizando, esta seção mostra que a utilização de CPU do IDD não depende do tamanho dos pacotes, mas sim do número de pacotes que passa pelo IDD, ou entrando ou saindo da máquina virtual. Conseqüentemente, a alocação de CPU no IDD pode ser determinada conhecendo-se as características do tráfego gerado pelas aplicações que executam nas máquinas virtuais. Além disso, mostramos que o IDD pode precisar de mais recursos para garantir que múltiplas máquinas virtuais simultaneamente

executando aplicações de rede não tenham o desempenho limitado pelo IDD.

6.3 Configurando um ambiente virtual

Nesta seção apresentamos uma análise de desempenho para diferentes configurações do ambiente virtual Xen. As configurações propostas visam obter uma melhor escalabilidade do IDD visto que este pode se tornar ponto de contenção de tráfego até mesmo para poucas máquinas virtuais executando aplicações que geram grande tráfego de rede.

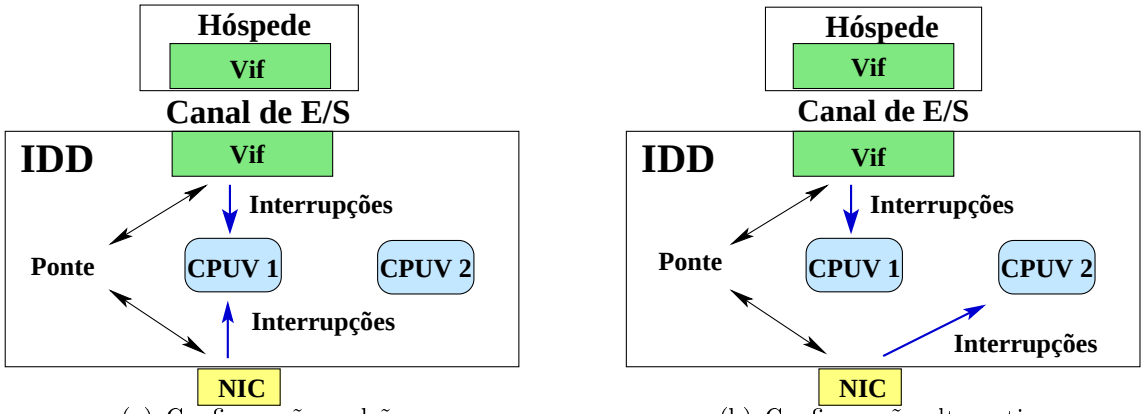
6.3.1 Dividindo a carga de um NIC em duas CPUs no IDD

Em seções anteriores discutimos que o IDD funciona como um intermediador de todo o tráfego causado pelas máquinas virtuais hóspedes. A interface de rede física (NIC) no IDD conecta a uma interface virtual (VIF - *Virtual Interface*), responsável pelos pacotes de uma certa máquina virtual. Os pacotes que chegam ao NIC e à VIF causam interrupções na CPU virtual do IDD, indicando que existem pacotes a serem processados.

Para o caso em que temos mais de uma CPU física no IDD, espera-se que as duas CPUs possam ser utilizadas de forma que o IDD possa atender o processamento de E/S de um número maior de máquinas virtuais. Entretanto, a configuração padrão do Xen associa todas as interrupções a apenas um dos processadores. Nesse caso, se tivermos uma situação em que o IDD é responsável pela contenção do tráfego de E/S e adicionarmos um novo processador no IDD, este continuará sendo o ponto de contenção de tráfego de rede do sistema e o processador adicionado não será utilizado. A figura 6.6(a) ilustra a situação em que temos um NIC e duas CPUs para o IDD. As interrupções geradas pelo NIC e pela VIF que se conecta à máquina virtual hóspede são todas associadas à primeira CPU.

Na tentativa de melhorar a utilização dos recursos disponíveis sem fazer nenhuma

modificação no Xen, modificamos a configuração dos IRQs (*Interrupt Request Line*) para o NIC e para a VIF. Um IRQ corresponde a um canal dedicado por onde um periférico pode lançar interrupções em um determinado processador. Tanto no Linux quanto no Xen, a configuração de IRQs pode ser modificada alterando-se uma máscara que indica as possíveis CPUs para os quais determinado dispositivo pode enviar interrupções. A figura 6.6(b) representa este segundo cenário, onde a configuração de associação dos IRQs foi modificada, enviando todas as interrupções geradas pelo NIC para a segunda CPU. Chamaremos o primeiro cenário de configuração padrão e o segundo de configuração alternativa.



(a) Configuração padrão (b) Configuração alternativa

Figura 6.6: Associação de interrupções de 1NIC em duas CPUs no IDD

Para comparar as duas configurações, avaliamos a taxa de processamento de rede e a utilização de CPU através do conjunto de testes de tráfego TCP. Utilizamos o Xencpu para medir utilização de CPU através da técnica de medição centralizada.

A figura 6.7 mostra um gráfico de *Kiviat* que compara, para os dois cenários de configuração, a utilização de CPU dos três processadores e a taxa de processamento de pacotes relativa ao limite máximo da interface de rede. Cada linha radial que parte do ponto central 0 representa uma destas métricas, tendo como valor máximo 1. A máquina virtual é executada em uma CPU diferente das CPUs do IDD e envia pacotes TCP de 1448 bytes o mais rápido possível para uma máquina real receptora de tráfego, conectadas por uma interface de 1 Gigabit.

Note que, no cenário em que utilizamos a configuração padrão, apenas a CPU 1 do IDD é utilizada. Já no cenário em que utilizamos a configuração alternativa existe processamento nas duas CPUs do IDD. Na CPU da máquina virtual podemos notar um pequeno aumento na utilização de CPU quando utilizamos a configuração alternativa. Entretanto, a taxa de processamento de pacotes TCP obtida com a configuração alternativa é um pouco menor do que a taxa de processamento obtida com a configuração padrão. Ou seja, existe mais trabalho sendo executado pelos processadores para adquirir uma taxa de processamento de rede menor (praticamente a mesma), o que é inesperado.

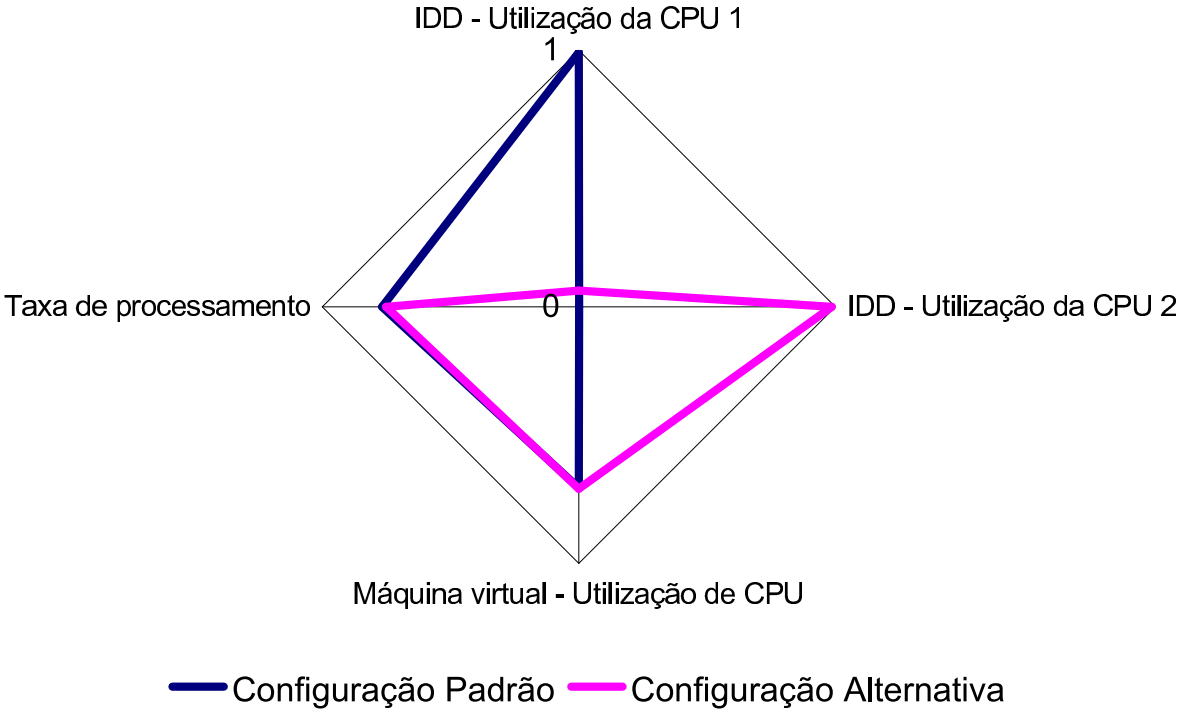


Figura 6.7: Avaliação de desempenho de duas configurações para um NIC e duas CPUs no IDD

O Xenoprof ainda não suporta SMP e por isso não foi utilizado nestes experimentos. Em trabalhos futuros, pretendemos desenvolver o suporte a SMP no Xenoprof e utilizá-lo para investigar as causas desta degradação na taxa de desempenho a um custo maior de utilização de CPU. Acreditamos que separar o processamento de E/S das duas interfaces em duas CPUs pode causar processamento extra devido à execução da ponte TCP que interliga os dois dispositivos. Entretanto, não descartamos que isso possa ser

uma falha no código do Xen. Além disso, queremos estudar o funcionamento do mecanismo de roteamento e verificar se ele pode ser utilizado para realizar balanceamento de carga nos processadores. Os cenários onde IRQs da VIF são associados à segunda CPU e os IRQs do NIC são associados à primeira CPU também foram avaliados e os resultados são semelhantes às da configuração alternativa. Quando associamos as interrupções do NIC e da VIF à segunda CPU ao mesmo tempo, não vemos resultados diferentes comparados com a configuração padrão.

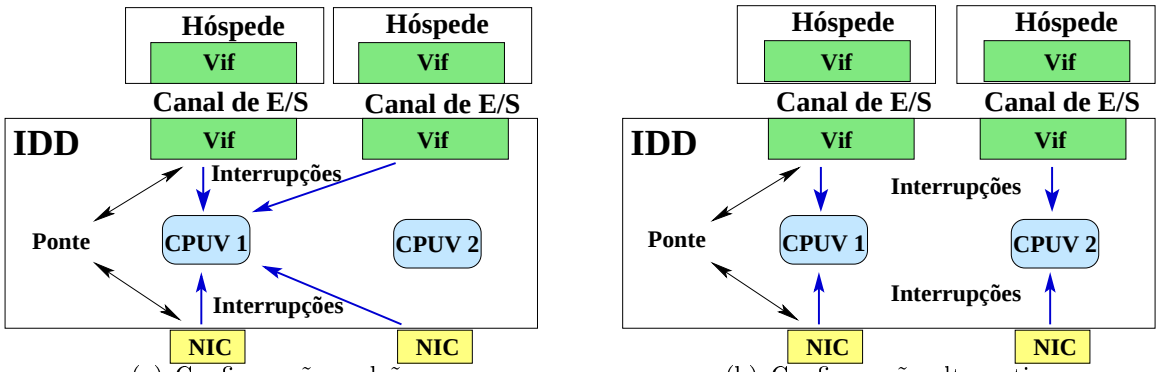
6.3.2 Dividindo a carga de dois NIC em duas CPUs no IDD

Na última seção mostramos que o processamento do tráfego de E/S causado por um NIC não pode ser dividido em mais de uma CPU na atual versão do Xen. Agora, considere um cenário onde temos mais de um NIC e mais de uma CPU no IDD. Novamente, a configuração padrão do Xen associa todas as interrupções causadas pelos NICs e pelas VIFs à primeira CPU, enquanto que as outras CPUs não executam nenhum processamento de E/S.

Vamos avaliar duas configurações para um cenário onde temos uma máquina com quatro CPUs e dois NICs. Configuramos o ambiente virtual com duas CPUs para o IDD e outras duas CPUs para duas máquinas virtuais. Cada máquina virtual executa em um processador e utiliza separadamente um dos NICs. A figura 6.8(a) representa a configuração padrão do Xen para este cenário, onde todas as interrupções geradas pelos NICs e VIFs são associadas à apenas uma das CPUs. A segunda CPU não é utilizada para processamento de tráfego de E/S, ficando o sistema limitado pelo processamento de um único processador no IDD. Da mesma forma que fizemos na seção passada, avaliamos uma configuração alternativa na tentativa de alocar melhor os recursos disponíveis. A figura 6.8(b) representa o segundo cenário, onde modificamos a configuração da associação de IRQs, enviando todas as interrupções relacionadas à segunda máquina virtual para a segunda CPU. Nesse cenário, todas as interrupções vindas do segundo NIC e da segunda VIF que se conecta na segunda máquina virtual

são associadas à segunda CPU na tentativa de balancear o processamento de tráfego de E/S.

Para a avaliação experimental da configuração alternativa, utilizamos as máquinas virtuais hóspedes rodando o conjunto de testes de tráfego TCP enviando pacotes de 1448 byte. Cada máquina virtual envia os pacotes para uma máquina cliente capaz de processar todos os pacotes sem se tornar ponto de contenção do sistema. Cada máquina virtual utiliza um NIC diferente para se comunicar com a máquina real receptora de tráfego.



(a) Configuração padrão (b) Configuração alternativa

Figura 6.8: Associação de interrupções de dois NICs em duas CPUs

A figura 6.9 mostra um gráfico de *Kiviat* que, para as duas configurações, compara a utilização de CPU das quatro CPUs e a taxa de processamento de pacotes obtida em cada máquina virtual relativa ao limite máximo do NIC. Novamente com a configuração padrão apenas a CPU 1 do IDD é utilizada e no cenário em que utilizamos a configuração alternativa existe processamento nas duas CPUs do IDD. Além disso, podemos notar um aumento marginal na utilização das duas máquinas virtuais quando utilizamos a configuração alternativa. Entretanto, as taxas de processamento de pacotes TCP obtidas com a configuração alternativa são menores do as taxas de processamento obtidas com a configuração padrão para as duas máquinas virtuais.

Novamente o uso do Xenoprof poderia explicar as razões deste resultado. Acreditamos que com o amadurecimento do código do Xen, várias soluções capazes de melhorar

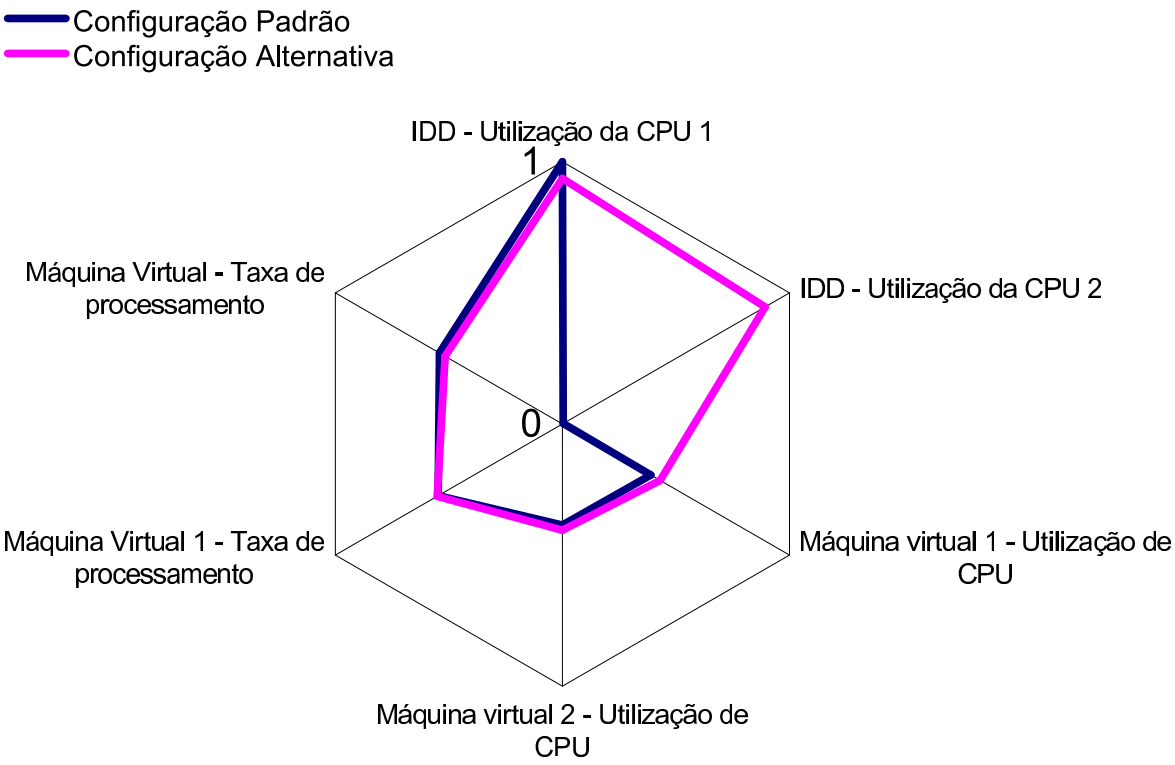


Figura 6.9: Avaliação de desempenho de duas configurações para dois NICs e duas CPUs no IDD

a escalabilidade do IDD ou até mesmo balancear o processamento de tráfego de E/S irão surgir. Como conclusão destes resultados podemos ver que a versão atual do Xen possui o processamento de tráfego de E/S limitado por uma única CPU no IDD, o que não é suficiente até mesmo para poucas máquinas virtuais executando aplicações que geram muito tráfego de rede. A implementação de múltiplos IDDs no Xen está ainda em desenvolvimento. O uso de mais de um IDD parece ser a solução imediata mais viável a ser adotada.

6.4 Entendendo Interferência entre Máquinas Virtuais

Finalmente, exploramos o impacto que a interferência entre máquinas virtuais compartilhando recursos físicos não particionáveis, pode causar no desempenho das aplicações executadas no sistema. O objetivo é investigar a existência de degradação de

desempenho quando temos mais de uma máquina virtual utilizando o mesmo processador e, conseqüentemente, estudar as causas desta degradação. Particularmente, investigamos o número de falhas ocorridas em memórias-cache. Avaliamos o desempenho da execução do conjunto de testes de CPU em três cenários diferentes. Para cada cenário medimos o desempenho de um número n de aplicações executadas simultaneamente, onde n varia de 1 a 5 nos experimentos. Todas as aplicações são sincronizadas para começar e terminar ao mesmo tempo. Os três cenários analisados são:

- **Linux**: O sistema Linux é utilizado como base de comparação para os resultados obtidos com o ambiente virtual Xen.

- **Mesmo Hóspede**: Neste cenário, as n aplicações são executadas simultaneamente em uma mesma máquina virtual. Este experimento é similar ao experimento realizado com o sistema Linux, porém, as aplicações executam em uma única máquina virtual.

- **Hóspedes diferentes**: Neste cenário, cada aplicação executa em uma máquina virtual hóspede diferente, todas na mesma CPU. O IDD executa em uma CPU separada. Como exemplo, para o experimento em que temos cinco aplicações, temos também cinco máquinas virtuais, cada uma executando uma das aplicações. Sendo assim, a cada vez que uma aplicação ocupa o processador, um número maior de instruções e dados passa pelo processador devido à execução do código do sistema operacional e da máquina virtual.

A figura 6.10 mostra o tempo gasto para executar de 1 a 5 compilações do código kernel para os três cenários descritos anteriormente. Esta figura mostra o tempo relativo ao Linux para a execução das aplicações como função do número de aplicações simultaneamente em execução. Comparando-se o cenário onde as aplicações executam na mesma máquina virtual, notamos que a diferença desse cenário em relação ao primeiro consiste basicamente no custo da própria virtualização. Entretanto, para o terceiro cenário, este desempenho relativo ao Linux piora à medida que aumentamos o

número de aplicações concorrentes, chegando a gastar cerca de 18% a mais de tempo do que o Linux para executar 5 aplicações.

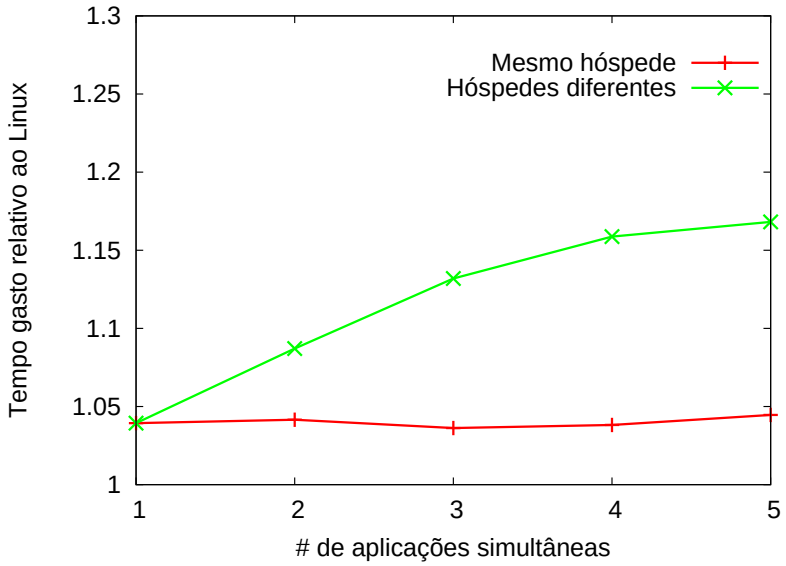


Figura 6.10: Tempo de execução relativo ao Linux para o conjunto de testes de CPU para três cenários

Para entendermos este efeito, utilizamos o Xenoprof e o Oprofile para coletar amostras de eventos de hardware para o Xen o Linux, respectivamente. Foram coletados o número de falhas nas caches L2 (*level 2*) e L3 (*level 3*) com um período de amostragem de 10^4 falhas. Estas caches estão organizadas em hierarquia de forma que se ocorrer uma falha na cache L1 o processador procura pela instrução ou dado na cache L2 e, da mesma forma, se ocorre uma falha na cache L2, a cache L3 é acessada.

Estes eventos de hardware são plotados relativos ao sistema Linux como uma função do número de aplicações concorrentes para o segundo e o terceiro cenários de experimentação. Estes resultados podem ser vistos na figura 6.11. Podemos notar que o aumento no tempo de execução relativo ao Linux quando aumentamos o número de máquinas virtuais é fortemente influenciado pelo aumento no número de falhas na cache L3. Quando uma única aplicação é executada e ocupa sozinha o processador, ela pode utilizar melhor a localidade temporal e espacial existente no código executado. Entretanto, quando acrescentamos mais aplicações concorrentes, uma aplicação polui as caches para as outras aplicações, com instruções e dados que não serão utilizados.

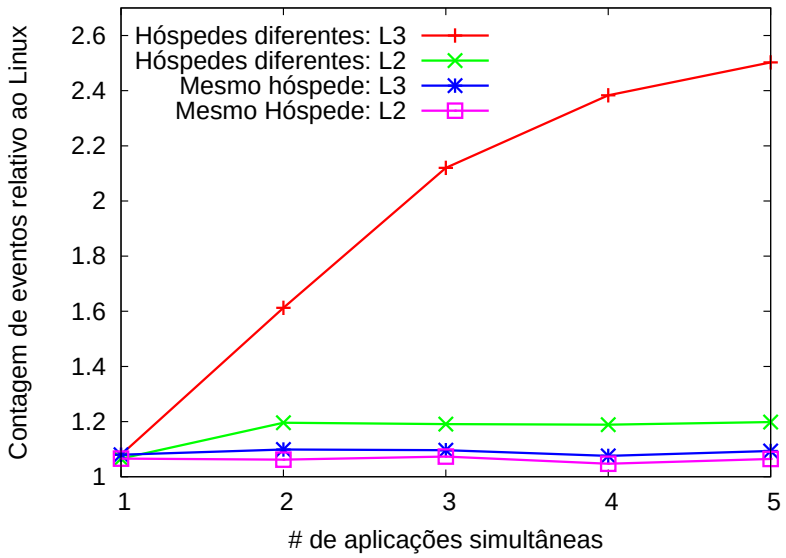


Figura 6.11: Contagem de falhas de cache relativa ao Linux para diferentes cenários de um ambiente virtual

Quando utilizamos aplicações executando em máquinas virtuais, cada vez que uma aplicação ocupa o processador, ela traz consigo para as caches uma quantidade de dados e código muito maior do que no cenário em que temos aplicações concorrentes em um sistema não virtual. Esta quantidade maior de código e dados é devido ao kernel do sistema operacional e códigos adicionais relativos à máquina virtual, o que polui os componentes compartilhados, causando um número maior de falhas nas caches e, conseqüentemente, uma maior degradação no desempenho. A cache L2 possui um pequeno aumento relativo ao Linux de uma para duas máquinas virtuais e, então, atinge seu limite superior para o número de falhas na cache. A cache do terceiro nível (L3) começa a estabilizar com cinco máquinas virtuais, causando uma piora no desempenho das aplicações que estão sendo executados de cerca de 18% comparado com o tempo de execução do mesmo número de aplicações concorrentes no sistema Linux.

Sumarizando, ao implantarmos aplicações em máquinas virtuais compartilhando os mesmo recursos, verificamos que as aplicações podem interferir no desempenho umas das outras. Nos experimentos esta degradação de desempenho em relação ao Linux chegou a cerca de 18% com 5 máquinas virtuais concorrentes. Esta queda no desempenho deve ser levada em consideração para a configuração de um ambiente virtual.

Capítulo 7

Conclusões e Trabalhos Futuros

Nesta dissertação apresentamos duas técnicas para medição e monitoramento de desempenho: (1) a centralizada no VMM que permite monitorar de uma forma simples e prática as máquinas virtuais sem conhecer detalhes do desempenho das aplicações que executam na máquina virtual e (2) a distribuída entre as máquinas virtuais que, apesar de mais complexa, permite acesso a contadores de hardware e informações sobre a execução das aplicações dentro das máquinas virtuais.

Nosso estudo de caso aborda as principais preocupações em termos de desempenho relativas à migração de aplicações de um ambiente real para um ambiente virtual. Para avaliarmos as duas técnicas de medição de desempenho apresentadas foi desenvolvida uma ferramenta chamada Xencpu, que utiliza a técnica centralizada para medir utilização de CPU no ambiente virtual Xen e utilizamos e avaliamos o desempenho do Xenoprof, uma ferramenta que utiliza a técnica descentralizada para obter informações do ambiente virtual. Como principais resultados obtidos no estudo de caso podemos sumarizar:

- o processamento de CPU para processar dados de E/S de rede é cerca de 3 a 5 vezes maior para uma máquina virtual do Xen quando comparado com o sistema Linux.
- a utilização de CPU do IDD não depende do tamanho dos pacotes, mas do

número de pacotes que entram e saem da máquina virtual, o que pode ser útil para estimar a alocação de CPU para o IDD baseado nas características do tráfego gerado pelas aplicações.

- a taxa de processamento de rede do IDD é limitada pela capacidade de processamento de uma única CPU, mesmo quando utilizamos mais de um processador no IDD.

- a degradação de desempenho causada por máquinas virtuais compartilhando o mesmo processador pode ser significativa e precisa ser contabilizada para a implantação de múltiplas máquinas virtuais compartilhando a mesma CPU.

De maneira geral, os resultados mostram diversas evidências de que o ambiente virtual Xen precisa de melhoras de desempenho, especialmente na arquitetura de E/S, para que possa ser utilizado para prover serviços que geram um grande tráfego de rede. Acreditamos que estes resultados possam guiar o processo de configuração de um servidor virtual e dirigir o desenvolvimento de soluções que melhorem o desempenho de sistemas virtuais e possibilitem a implantação de aplicações em servidores que utilizam estes ambientes.

Como direções para trabalhos futuros, pretendemos desenvolver soluções capazes de diminuir a degradação de desempenho causada pela interferência entre máquinas virtuais compartilhando os mesmos recursos físicos. Além disso, planejamos desenvolver modelos analíticos para prever o comportamento de aplicações em ambientes virtuais e conseqüentemente desenvolver uma ferramenta de configuração automática para um servidor virtual.

Referências Bibliográficas

[1] A. Awadallah and M. Rosenblum. The vMatrix: Server Switching. In *Proc. of IEEE 10th International Workshop on Future Trends in Distributed Computing Systems (IEEE FTDCS 2004)*, Suzhou, China, Maio 2004.

[2] B. Abrahao, V. Almeida, J. Almeida, A. Zhang, D. Beyer, and F. Safai. Self-Adaptive SLA-Driven Capacity Management for Internet Services. In *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Vancouver, Canadá, Abril 2006.

[3] M. Allman, S. Floyd, and C. Partridge. Increasing TCP's Initial Window. RFC 3390 (Proposed Standard), Outubro 2002.

[4] A.Moshchuk, T. Bragin, S. Gribble, and H. Levy. A Crawler-based Study of Spyware on the Web. In *Proc. of 13th Annual Network and Distributed System Security (NDSS)*, San Diego, CA, Fevereiro 2006.

[5] Apache. http://httpd.apache.org.

[6] Y. Bard. Performance Analysis of Virtual Memory Time-Sharing Systems. *Proc. of IBM Systems Journal*, 14(4):366–384, Setembro 1975.

[7] Y. Bard. An analytic Model of the VM/370 System. *IBM Journal of Research and Development*, 22(5):498–508, Setembro 1978.

[8] Y. Bard. The VM/370 Performance Predictor. *ACM Computer Surveys*, 10(3):333–342, 1978.

[9] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the Art of Virtualization. In *Proc. of 19th ACM Symposium on Operating Systems Principles*, Outubro 2003.

[10] F. Benevenuto, C. Fernandes, M. Santos, V. Almeida, J. Almeida, G. Janakiraman, and J. R. Santos. Performance Models for Virtualized Applications. In *Proc. of 1st Workshop on XEN in High-Performance Cluster and Grid Computing (XHPC'06)*, Sorrento, Itália, Dezembro 2006.

[11] L. Cherkasova and R. Gardner. Measuring CPU Overhead for I/O Processing in the Xen Virtual Machine Monitor. In *Proc. of USENIX Annual Technical Conference*, Abril 2005.

[12] D. Ardagna and C. Francalanci. A Cost-Oriented Approach for the Design of IT Architectures. In *Journal of Information Technology*, Fevereiro 2005.

[13] S. Devine, E. Bugnion, and M. Rosenblum. Virtualization System Including a Virtual Machine Monitor for a Computer with a Segmented Architecture. Technical Report US Patent 6397242, vmware, Outubro 1998.

[14] S. Eranian. Enhanced Virtualization on Intel Architecture-based Servers. Technical Report Intel Solutions White Paper, Intel, Julho 2005.

[15] Ferramenta TTCP. http://www.pcusa.com/utilities/pcattcp.htm.

[16] D. Ferrari, G. Serazzi, and A. Zeigner. Measurement and Tuning of Computer Systems. In *International CMG Conference*, pages 793–794, Dezembro 1984.

[17] R. Figueiredo, P.A. Dinda, and J. Fortes. Resource Virtualization Renaissance. In *Proc. of IEEE Internet Computing*, Maio 2005.

[18] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. Grid Services for Distributed System Integration. *Computer*, 35(6), 2002.

[19] K. Fraser, S. Hand, R. Neugebauer, I. Pratt, A. Warfield, and M. Williamson. Reconstructing I/O. Technical Report UCAM-CL-TR-596, Cambridge University, Agosto 2004.

[20] K. Fraser, S. Hand, R. Neugebauer, I. Pratt, A. Warfield, and M. Williamson. Safe hardware Access with the Xen Virtual Machine Monitor. In *Proc. of 1st Workshop on Operating System and Architectural Support for the on demand IT InfraStructure (OASIS)*, Outubro 2004.

[21] D. Gupta, R. Gardner, and L. Cherkasova. XenMon: QoS Monitoring and Performance Profiling Tool. Technical Report HPL-2005-187, HP Labs, Outubro 2005.

[22] Intel Vanderpool Technology. http://www.intel.com/technology/computing/vptech/.

[23] M. Kallahalla, M. Uysal, R. Swaminathan, D. E. Lowell, M. Wray, T. Christian, N. Edwards, C. I. Dalton, and F. Gittler. SoftUDC: A Software Based Data Center for Utility Computing. In *Journal of IEEE Computer*, pages 46–54, Novembro 2004.

[24] J. LeVasseur, V. Uhlig, J. Stoess, and S. Gotz. Unmodified Device Driver Reuse and Improved System Dependability Via Virtual Machines. In *Proc. of Operating Systems Design and Implementation (OSDI)*, Dezembro 2004.

[25] D. Magenheimer and T. Christian. vBlades: Optimized Paravirtualization for the Itanium Processor Family. In *Proc. of USENIX Virtual Machine Research and Technology Symposium (VM)*, San Jose, CA, Maio 2004.

[26] S. Malaika, A. Eisenberg, and J. Melton. Standards for databases on the grid. *SIGMOD Record*, 32(3), 2003.

[27] D. Menasce, V. Almeida, and L. Dowdy. *Capacity Planning and Performance Modeling: From Mainframes to Client-Server Systems.* Prentice-Hall, Inc., Upper Saddle River, NJ, EUA, 1994.

[28] D. Menascé. Virtualization: Concepts, Applications, and Performance. In *Proc. of Computer Measurement Group's 2005 International Conference (CMG)*, Orlando, Fl, Dezembro 2005.

[29] A. Menon, J. R. Santos, Y. Turner, G. Janakiraman, and W. Zwaenepoel. Diagnosing Performance Overheads in the Xen Virtual Machine Environment. In *Proc. of First ACM/USENIX Conference on Virtual Execution Environments (VEE'05)*, Chicago, IL, Junho 2005.

[30] V. Misra, W. Gong, and D. Towsley. Stochastic Differential Equation Modeling and Analysis of TCP-Windowsize Behavior. In *Proc. of Performance*, Istanbul, Turquia, Outubro 1999.

[31] D. Mosberger and T. Jin. httperf: A Tool for Measuring Web Server Performance. In *Proc. of First Workshop on Internet Server Performance*, pages 59—67, Madison, WI, Junho 1998.

[32] E. Nahum. Deconstructing SPECweb99. In *Proc. of 7th International Workshop on Web Content Caching and Distribution (WCW)*, Boulder, Colorado, EUA, Agosto 2002.

[33] OProfile. http://oprofile.sourceforge.net.

[34] PlanetLab. An Open Platform for Developing, Deploying, and Accessing Planetary-Scale Services. Available at: http://www.planet-lab.org/, Março 2002.

[35] QEMU. http://fabrice.bellard.free.fr/qemu/.

[36] C. Strachey. Time Sharing in Large, Fast Computers. In *Proc. of IFIP Congress*, pages 336–341, Outubro 1959.

[37] J. Sugerman, G. Venkitachalam, and B. Lim. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor. In *Proc. of USENIX Annual Technical Conference*, Junho 2001.

[38] UML - User mode Linux. http://user-mode-linux.sourceforge.net/.

[39] Virtual PC. http://www.microsoft.com/windows/virtualpc/default.mspx.

[40] VMWare Consolidation. http://www.vmware.com/solutions/consolidation/index.html.

[41] VMWare Web Site. http://www.vmware.com.

[42] M. Vrable, J. Ma, J. Chen, D. Moore, E. Vandekieft, A. C. Snoeren, G. M. Voelker, and S. Savage. Scalability, Fidelity, and Containment in the Potemkin Virtual Honeyfarm. *SIGOPS Operating Systems Review*, 39(5):148–162, 2005.

[43] vServers. http://linux-vserver.org/.

[44] C. Waldspurger. Memory Resource Management in VMware ESX Server. In *Proc. of Operating Systems Design and Implementation (OSDI)*, Dezembro 2002.

[45] A. Whitaker, R. Cox, M. Shaw, and S. Gribble. Constructing Services with Interposable Virtual Hardware. In *Proc. of Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, Março 2004.

[46] A. Whitaker, R. Cox, M. Shaw, and S. Gribble. Rethinking the Design of Virtual Machine Monitors. *Journal of IEEE Computer*, 38(5):57–67, Maio 2005.

[47] A. Whitaker, M. Shaw, and S. Gribble. Scale and Performance in the Denali Isolation Kernel. In *Proc. of Operating Systems Design and Implementation (OSDI)*, Boston, MA, Dezembro 2002.

[48] J. Wilkes, J. Mogul, and J. Suermondt. Utilification. In *Proc. of the 11th ACM SIGOPS European Workshop*, Setembro 2004.

[49] Xenoprof. http://xenoprof.sourceforge.net.