VEPMon: Uma Ferramenta de Monitoração de Desempenho para Ambientes Virtuais

Matheus Santos[1], **César Fernandes**[1], **Fabrício Benevenuto**[1], **Virgílio Almeida**[1], **Jussara Almeida**[1]

[1] Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)
Av. Antônio Carlos, 6627 - Pampulha - Belo Horizonte - MG
CEP 31270-901 Fone: +5531 3409.5860 Fax: +5531 3409.5858

{mtcs, cesar, fabricio, virgilio, jussara}@dcc.ufmg.br

Abstract. *This work presents VEPMon, a performance monitoring tool for virtual machines executing on the Xen virtual environment. Our tool focuses on calculating metrics which can be used as input for analytic models to predict performance of Web services. Moreover, our tool provides a flexible architecture, with low system intrusion. As a proof of its functionality, we present a case study in which we create an analytic model based on queuing theory to predict response time of an HTTP server and VEPMon is used to measure the input metrics to our model.*

Resumo. *Este trabalho apresenta o VEPMon, uma ferramenta para monitorar o desempenho de máquinas virtuais em execução no ambiente virtual Xen. Nossa ferramenta tem o enfoque no calculo de métricas que podem ser utilizadas como entradas para modelos analíticos para prever desempenho de serviços Web. Além disso, nossa ferramenta possui uma interface compacta e uma arquitetura flexível e de baixa intrusão no sistema. Como prova de sua funcionalidade, apresentamos um estudo de caso no qual criamos um modelo analítico baseado em teoria de filas para prever o tempo de resposta de um servidor HTTP e o VEPMon é utilizado para coletar as métricas de entrada do modelo.*

1. Introdução

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 infra-estrutura, reduzindo custos de gerenciamento e permitindo a criação de um ambiente capaz de se adaptar a mudanças na carga das aplicações.

Neste contexto, monitorar o desempenho de aplicações nesses ambientes é uma tarefa essencial para a implantação de qualquer serviço em plataformas virtualizadas. Existem várias ferramentas propostas [Gupta et al. 2005, Xenoprof] que monitoram várias métricas importantes como, por exemplo, utilização de CPU e informações sobre disco e memória. Entretanto, essas ferramentas não fornecem todas as métricas necessárias para a criação de modelos analíticos, essenciais para o planejamento da capacidade das máquinas virtuais (VMs) [Menasce et al. 2004, Benevenuto et al. 2006] e

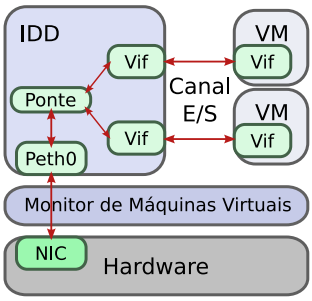

Figura 1. Arquitetura do Xen

também para sistemas auto-adaptativos de alocação de recursos para serviços virtualizados [Abrahao et al. 2006, Cunha et al. 2008].

Este trabalho apresenta o VEPMon (Virtual Environment Performance Monitor), uma ferramenta para monitorar o desempenho de máquinas virtuais em execução no ambiente virtual Xen [Barham et al. 2003]. Nossa ferramenta tem o enfoque na monitoração de serviços Web, calculando métricas que podem ser utilizadas como entradas para modelos analíticos. Como uma forma de validar o funcionamento da nossa ferramenta, apresentamos um estudo de caso no qual criamos um modelo analítico baseada em teoria de filas para prever o tempo de resposta de um servidor HTTP. O VEPMon é utilizado para medir as métricas de entrada do modelo. Além disso, apresentamos uma análise da sobrecarga que nossa ferramenta causa no sistema.

O restante deste trabalho está organizado da seguinte forma. Na próxima seção discutimos trabalhos relacionados e aspectos da arquitetura do Xen necessários para o entendimento do trabalho. A seção 3 apresenta o VEPMon, discute sua arquitetura, métricas calculadas e apresenta suas funcionalidades. A seção 4 provê um estudo da sobrecarga da ferramenta. A seção 5 apresenta o estudo de caso no qual utilizamos a ferramenta e, finalmente, a seção 6 conclui o artigo e apresenta direções para trabalhos futuros.

2. Trabalhos Relacionados e o Ambiente Virtual Xen

Esta seção apresenta uma rápida descrição dos aspectos do Xen necessários ao entendimento do trabalho e discute trabalhos relacionados.

2.1. Arquitetura do Xen

Xen é um monitor de máquinas virtuais (VMM) que permite múltiplas instâncias de sistemas operacionais executarem concorrentemente em uma única máquina física [Barham et al. 2003]. O Xen utiliza paravirtualização, onde o VMM pode ser acessado através de uma máquina virtual levemente modificada em relação ao hardware. A figura 1 mostra a arquitetura do Xen. Cada aplicação executando em um SO acessa dispositivos de hardware através de uma VM especial e com acessos privilegiados ao hardware chamada IDD (*isolated driver domain*), também chamado de Dom0. As outras VMs executam dispositivos simplificados que se comunicam com o IDD para acessar os verdadeiros dispositivos de hardware. Uma VM acessa o hardware indiretamente através de um dispositivo virtual conectado ao IDD. Para evitar cópia de dados, referências às páginas são transferidas através deste dispositivo ao invés dos verdadeiros dados de entrada e saída [Fraser et al. 2004].

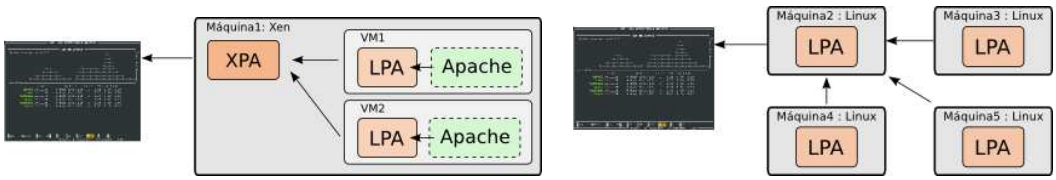

Figura 2. Possíveis organizações da ferramenta

2.2. Trabalhos Relacionados

A idéia de uma ferramenta de monitoramento para o Xen foi inicialmente abordada através de uma ferramenta chamada Xenmon [Gupta et al. 2005]. O Xenmon foi desenvolvido para monitorar utilização de CPU de máquinas virtuais no ambiente virtual Xen e estimar a utilização de CPU no IDD devido a operações de E/S de cada VM. De uma maneira simplificada, o Xenmon estima o custo de CPU para cada evento de E/S entre VMs e o IDD e contabiliza o número de eventos de E/S de cada VM para calcular a fração de tempo da CPU do IDD consumida com cada VM.

A abordagem utilizada no Xenmon é similar à nossa abordagem de caixa preta. Outra ferramenta similar à nossa abordagem de caixa preta é a ferramenta xm top, que possui funções semelhantes às do comando top no Linux. Como diferencial em relação ao xm top e do Xenmon, nossa ferramenta possui o monitoramento de caixa cinza.

Outra importante ferramenta de monitoramento para o Xen é o Xenoprof [Menon et al. 2005]. O Xenoprof coleta estatísticas de eventos de hardware de ambientes virtuais do Xen tais como falhas nas caches, falhas na TLB, número de instruções executadas, etc. Nossa ferramenta e o Xenoprof possuem objetivos diferentes e, conseqüentemente, se complementam.

3. VEPMon

O VEPMon é uma ferramenta de monitoração de desempenho voltada para ambientes virtuais e Linux. Ela realiza a leitura temporizada dos registros de estatísticas do *kernel* e utiliza valores médios para exibir suas métricas de desempenho. Atualmente, o VEPMon está disponível somente para o ambiente virtual Xen e para o Linux, ambas em modo texto. Nesta seção apresentamos a arquitetura da ferramenta, suas métricas e funcionalidades.

3.1. Arquitetura

O VEPMon é composto por dois tipos de módulos: a interface de monitoração e os agentes de desempenho. Para esta versão do VEPMon foram desenvolvidos dois agentes de desempenho: o LPA (*Linux Performance Agent*) e o XPA (*Xen Performance Agent*). Esses agentes executam na máquina a ser monitorada esperando pela conexão da interface, que pode estar localizada em outra máquina já que a comunicação é feita via TCP.

O protocolo de comunicação adotado possibilita a múltipla conexão entre módulos, permitindo uma maior flexibilidade da ferramenta. Um módulo de desempenho, LPA ou XPA, pode se conectar a outro módulo de desempenho, fazendo o segundo como um intermediário. Essa característica permite monitoramento de métricas obtidas junto ao sistema operacional em execução dentro da VM ou mesmo em um conjunto de máquinas Linux, conforme ilustrado na figura 2.

Essa flexibilidade na comunicação entre os módulos permite ao VEPMon funcionar de acordo com duas abordagens em ambientes virtuais chamadas de *caixa preta* e

caixa cinza. Na abordagem *caixa preta* é possível obter as métricas de desempenho das máquinas virtuais executando o agente de monitoração de desempenho, *XPA*, no Dom0 do Xen. Nesse caso, as métricas capturadas ficam restritas às métricas providas pelo ambiente virtual. No modo *caixa cinza* é possível extrair todas as métricas capturadas pelo *LPA* das máquinas virtuais. Isso é feito através da execução de um agente de monitoração em cada máquina virtual. Assim, cada máquina virtual é tratada como uma máquina real no ambiente de monitoração.

O VEPMon foi projetado inicialmente para monitoração, entretanto ele também realiza a medição do desempenho do ambiente durante a execução de uma determinada tarefa, especificada com um comando. Através da interface pode-se requisitar o modo medição e especificar a tarefa a ser monitorada, de forma que ao final da execução da tarefa a ferramenta produz o cálculo das métricas. Esse modo de operação é muito útil para a automatização de experimentos, como os que realizamos no nosso estudo de caso.

A comunicação entre a interface e os módulos é feita por rede TCP. O início do processo de medição é feito pela interface, que desempenha o papel de servidor. Após os agentes de desempenho se conectarem à interface, ela envia para os agentes de desempenho o modo de operação, que pode ser medição ou monitoramento, e os parâmetros a serem seguidos. Se o modo de operação for medição, a interface envia aos agentes o comando a ser executado como parâmetro, que depois de executá-lo enviam a medição para a interface. Já no modo de monitoração, as métricas são enviadas de tempos em tempos pelos agentes, logo após serem lidas do sistema operacional.

3.2. Métricas

As métricas coletadas pelo VEPMon são agrupadas nos seguintes recursos: CPU, memória, rede, disco, sistema, máquinas virtuais e serviço, de acordo com a tabela 1. Grande parte das métricas coletadas são métricas típicas do Linux e possuem definição simples e suporte em praticamente todas as versões mais novas do *kernel*. Entretanto, algumas métricas como os tempos de interrupção de software e hardware dependem da versão e do tipo de sistema nativo.

Em máquinas SMP, é possível exibir as métricas de CPU para cada processador separadamente utilizando-se o modo SMP da ferramenta. As métricas de serviço Web são ativadas apenas se o serviço suportado pela ferramenta estiver ativo e configurado corretamente. Atualmente, as métricas de serviço são obtidas através da interação da ferramenta com o servidor Web Apache [Apache], em execução no Linux ou em ambientes virtuais. Como veremos na seção 5, a demanda e a utilização de um recurso são métricas muito úteis na construção de modelos analíticos.

3.3. Interface

A interface da ferramenta foi projetada para a exibição compacta de várias máquinas simultaneamente em forma de lista. Ela é dividida em seções relativas ao recurso monitorado que são CPU, memória, rede, disco, sistema, máquinas virtuais e serviço Web. As seções são selecionadas pela letra chave ou através do mouse no menu inferior. A tela de sistema possui, além das suas métricas, um gráfico de escala adaptativa que exibe a média das cargas médias das máquinas monitoradas em relação ao instante de captura das cargas.

	Métricas de CPU		**Métricas de Sistema**
Usr	Tempo em processo de Usuário	Time	Hora local da máquina
Nic	Tempo de Nice	Up Time	Tempo total de funcionamento
Sys	Tempo em processo de sistema	Users	Número de usuários logados
Idl	Tempo ocioso	Average Load	Carga média da carga de sistema em 5, 10 e 15 minutos
IOW	Tempo bloqueado esperando E/S		
HI	Tempo de interrupção de hardware		**Métricas de Máquinas Virtuais**
		Domain Name	Nome da máquina virtual
SI	Tempo de interrupção de software	ID	Identificador da máquina virtual
		CPU	Utilização de CPU
Stl	Tempo steal do Linux	Mem	Memória disponível
		NetRX	KBytes de dados recebidos por rede
	Métricas de Memória	NetTX	KBytpes de dados transmitidos por rede
Total	Total de memória disponível		
Used	Memória sendo utilizada		**Métricas de Rede**
Free	Memória livre	Name	Nome da interface de rede
Buff	Memória em buffers do *kernel*	PacketsRX	Número de pacotes recebidos
Cache	Memória utilizada como cache	PacketsTX	Número de pacotes transmitidos
Swap	Memória de troca disponível	ThroughputRX	Mb de dados recebidos por segundo
SwUsed	Memória de troca utilizada	ThroughputTX	Mb de dados transmitidos por segundo
	Métricas de Serviço		**Métricas de Disco**
Req	Número de requisições HTTP	Name	Nome do dispositivo de disco
Req.KB	Total de dados das requisições HTTP em KBytes	Reads	Número de requisições de leitura
		Writes	Número de requisições de escrita
ReqRate	Requisições atendidas/s	Util	Tempo ocupado do disco
CPUDem	Demanda de CPU		
DiskDem	Demanda de Disco		

Tabela 1. Descrição das métricas coletadas pela ferramenta

4. Sobrecarga e Testes do VEPMon

Tanto para a realização do estudo de caso quanto para o estudo da sobrecarga do VEPMon utilizamos o ambiente experimental descrito a seguir.

Utilizamos um servidor Intel com dois processadores Xeon de 64 bits e 3.2 GHz com 2 GB de RAM, um disco de 7200 RPM e 8MB de cache e duas placas GBit Ethernet. Conectado diretamente com o servidor está um Athlon64 de 3GHz com 2 GB de RAM e duas placas de rede GBit Ethernet. Utilizamos o Xen 3.0.4 com arquitetura i386, tanto no IDD, quanto na VM e na máquina cliente, com distribuição Linux Debian Etch 1386 com *kernel* 2.6.12. Tanto a VM, quanto o IDD, e a máquina clientes tiveram a memória limitada em 512MB.

Os testes foram realizados com o objetivo de garantir a precisão nos dados reportados. Dessa forma, a medida que cada funcionalidade foi implementada, um teste foi realizado para determinar a corretude da coleta do dado em questão.

Para medir a sobrecarga da ferramenta, utilizamos a compilação de um *kernel* dentro de um VM com o VEPMon em execução e sem o VEPMon em execução. Com o VEPMon em execução realizamos experimentos com vários intervalos de monitoração, variando de 1 a 30 segundos. O desempenho da ferramenta foi medido com scripts que capturam o tempo em que a CPU ficou ocupada através do */proc* no Linux e através de uma ferramenta muito simples, que desenvolvemos anteriormente, chamada XenCPU [Benevenuto et al. 2006]. A interferência da ferramenta foi medida dividindo-se o tempo em que a CPU ficou ocupada utilizando-se o VEPMon ao compilar o *kernel* pelo tempo em que a CPU ficou ocupada sem a execução do VEPMon.

Para todos os intervalos analisados a interferência da ferramenta foi menor que 0,1%. Pelo fato da ferramenta usar rede TCP para a comunicação entre os módulos, notamos que, para cada agente de desempenho executando em uma máquina virtual ou em uma máquina linux, é consumido $10kbits$ a cada vez que as medidas são enviadas para a interface para cada agente de desempenho. Portanto, se em um ambiente virtual a monitoração

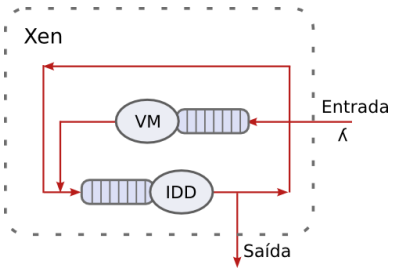

Figura 3. Representação do Modelo de Filas para o ambiente Xen

é feita a cada 5 segundos, com um agente de desempenho por VM, a ferramenta consome $2kbits/s$ por VM.

5. Estudo de Caso

Estimar o tempo de resposta médio de aplicações em execução em máquinas virtuais pode ser bastante útil para manter acordos do nível do serviço (SLA) estabelecidos ainda no ambiente real ou mesmo para permitir que o ambiente virtual maximize a alocação de recursos do ambiente virtual de forma a manter o SLA das aplicações em execução.

Nesta seção, apresentamos um estudo de caso com o objetivo de demonstrar uma possível situação em que as métricas coletadas pela ferramenta poderiam ser utilizadas. Para tal, introduziremos um modelo analítico baseado em teoria de filas, cujos valores de entrada foram coletados pela ferramenta, para estimar o tempo de resposta médio de um servidor HTTP de conteúdo estático e então mostramos resultados experimentais que validam o modelo. Nosso modelo é aberto e de valores médios. Além disso, como premissas do nosso modelo consideramos que o ambiente virtual possui uma VM e duas CPUs, uma alocada para uma VM e a outra alocada para o IDD. Não consideramos disco no modelo por simplicidade, já que a carga utilizada não gera muita atividade de disco. A figura 3 mostra a representação do modelo para o ambiente virtual descrito abaixo. Uma descrição detalhada sobre teoria de filas pode ser encontrada em [Menasce et al. 2004].

5.1. Modelo de Tempo de Resposta

Para definir o tempo de resposta, precisamos inicialmente definir três coisas: 1) demanda média por serviço no recurso k, D_k, onde k representa um recurso alocado pela VM ou pelo IDD como CPU, disco, etc. 2) a utilização média do recurso k, U_k e 3) o tempo de residência médio no recurso k, R_k. Note que neste modelo, como representado na figura 3, somente CPU da VM e CPU do IDD foram incluídos.

D_k é o tempo que uma requisição necessita para ser atendida pelo recurso k. D_k pode ser calculada diretamente dividindo-se o tempo que o recurso k ficou ocupado pelo numero de requisições completadas pelo sistema em um determinado momento. De fato, nossa ferramenta calcula D_k a partir do número de requisições completadas, monitoradas diretamente nos logs do apache [Apache].

U_k pode ser entendida como a proporção do tempo que o recurso k ficou ocupado durante um intervalo de tempo. A ferramenta coleta a U_k para as CPUs da VM e do IDD.

O tempo de residência médio é o tempo total de uma requisição no recurso k, incluindo o tempo gasto esperando na fila pelo recurso. R_k para um modelo aberto pode ser obtido através de uma equação simples descrita abaixo:

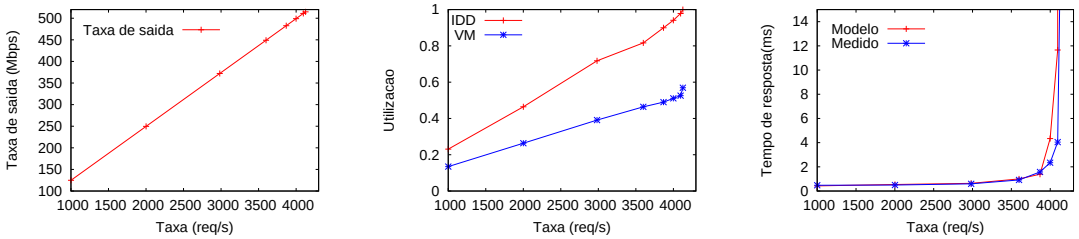

Figura 4. Taxa de Saída(esq.), Utilzação de CPU(cent.) e Tempo de resposta dos experimentos com o servidor HTTP(dir.)

$$R_k = \frac{D_k}{1 - U_k} \tag{1}$$

O tempo de resposta médio de uma requisição corresponde à soma do tempos de residência em todos os recursos por onde as requisições passam, nominalmente CPU da VM e CPU do IDD.

5.2. Análise do Desempenho

Para avaliarmos um servidor HTTP de forma experimental, utilizamos como clientes o httperf [Mosberger and Jin"1998] e como servidor Web o Apache [Apache] versão 2.0.55. O httperf é uma ferramenta que permite gerar várias requisições HTTP e medir o desempenho do servidor do ponto de vista dos clientes. O httperf é executado na máquina cliente, enviando requisições ao servidor, medindo a taxa de saída de requisições e o tempo de resposta. Uma VM hospeda o servidor apache em uma única CPU e o IDD executa em outra CPU separadamente. As duas cargas de trabalho utilizadas pelos clientes e o conteúdo do servidor foram geradas pelo SPECWeb99 [SPEC].

Podemos ver pelo gráfico na figura 4 (esq.), que a banda utilizada cresce linearmente com o aumento da taxa de requisições. Podemos notar que o ponto de contenção do sistema não é a rede, já que sua capacidade é de 1 GBit e sua utilização máxima é menor que 550 MBits/s. Observando gráfico 4(cent.) vemos que o sistema começa a saturar por volta de 4000 requisições por segundo onde a utilização do CPU do IDD chega a 100%. Vale ressaltar que a utilização do IDD chega a 100% devido ao fato de ele precisar emular parte da pilha TCP/IP, bem como pela necessidade de encaminhar cada pacote à VM de destino. Como podemos ver na figura 4 (dir.), a curva do tempo de resposta previsto pelo modelo acompanha a curva do tempo de resposta medida com o httperf.

6. Conclusões e Trabalhos Futuros

Este trabalho apresenta o VEPMon (*Virtual Environment Performance Monitor*), uma ferramenta para monitorar o desempenho de máquinas virtuais em execução no ambiente virtual Xen [Barham et al. 2003]. Através de um estudo de caso mostramos que o VEPMon pode ser útil para a obtenção de métricas de entrada para modelos analíticos. Além disso, mostramos que o VEPMon é pouco intrusivo no desempenho do sistema.

Acreditamos que o VEPMon é uma ferramenta importante e útil para o monitoramento de serviços Web virtualizados que necessitam de planejamento de capacidade e alocação de recursos para o ambiente virtual. Como trabalhos futuros pretendemos estender a ferramenta para monitorar outros ambientes como o VMWare e o HP-UX. Além disso, visamos a implementação de uma interface gráfica para o VEPMon, a inclusão de estatísticas de banco de dados e a implementação de históricos de dados.

7. Agradecimentos

Este trabalho foi desenvolvido em colaboração com a HP Brasil P&D.

Referências

Abrahao, B., Almeida, V., Almeida, J., Zhang, A., Beyer, D., and Safai, F. (2006). Self-Adaptive SLA-Driven Capacity Management for Internet Services. In *IEEE/IFIP NOMS*, Vancouver, Canada.

Apache. http://httpd.apache.org.

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

Benevenuto, F., Teixeira, C., Caldas, M., Almeida, V., Almeida, J., Santos, J. R., and Janakiraman, G. (2006). Performance Models for Applications on Xen. In *Proc. of XHPC'06*, volume LNCS 4331, Sorrento, Itália. Springer-Verlag.

Cunha, Í., Viana, I., Palotti, J., Almeida, J., and Almeida, V. (2008). Analyzing Security and Energy Tradeoffs in Autonomic Capacity Management. In *IEEE/IFIP NOMS*, Salvador, Brazil.

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

Gupta, D., Gardner, R., and Cherkasova", L. (2005). XenMon: QoS Monitoring and Performance Profiling Tool. Technical Report HPL-2005-187, HP Labs.

Menasce, D. A., Dowdy, L. W., and Almeida, V. A. F. (2004). *Performance by Design: Computer Capacity Planning By Example*. Prentice Hall PTR, Upper Saddle River, NJ, USA.

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

Mosberger, D. and Jin", T. (1998). httperf: A Tool for Measuring Web Server Performance. In *Proc. of Workshop on Internet Server Performance*, Madison, WI.

SPEC. `http://www.spec.org`.

Xenoprof. http://xenoprof.sourceforge.net.