Análise Quantitativa de Técnicas de Virtualização Como Ambiente de Testes

Artur Baruchi
IBM Brasil – Software Group
IBM Brasil, São Paulo
abaruchi@br.ibm.com

Ricardo L. Piantola
IBM Brasil – Software Group
IBM Brasil, São Paulo
piantola@br.ibm.com

ABSTRACT

Com o ressurgimento da Virtualização, uma das suas principais aplicações é a criação de ambientes de testes e homologação de sistemas. Isso se deve principalmente às características que um ambiente virtualizado deve possuir, como encapsulamento e isolamento. Entretanto, existem, nos dias atuais, diversas tecnologias de virtualização, como a paravirtualização e a virtualização completa, cada uma com características específicas que podem beneficiar ou prejudicar determinados tipos de testes. Neste trabalho serão abordadas algumas das principais técnicas de virtualização e como o uso de uma determinada técnica pode influenciar os resultados finais de um teste de software. Para mensurar e avaliar as principais técnicas de virtualização foi utilizado o monitor de máquinas virtuais Xen e um *benchmark* de compilação para verificar a sobrecarga imposta pela camada de software adicional existente em quase todos os sistemas virtualizados. A análise dos resultados demonstrou que a paravirtualização possui desempenho muito próximo ao de uma máquina tradicional (6,5% pior para o teste executado) sendo esta a melhor opção para a realização de testes de sistemas. Já ambientes que utilizam virtualização completa apresentam um desempenho inferior (68,5% pior para o teste executado), mas que é compensado por manter o código fonte do Sistema Operacional inalterado.

Palavras-chave

Teste de software, Virtualização, Sistemas Operacionais, Monitor de Máquinas Virtuais.

1. Introdução

Apesar do conceito de Máquinas Virtuais (MV) ter surgido no início dos anos 70 [1], a virtualização começou a popularizar-se há pouco tempo. Essa popularidade deve-se, em grande parte, ao barateamento de componentes de *Hardware*, como memória, e ao advento de processadores *Multi-Core*. Com esses eventos, tornou-se bastante comum encontrar equipamentos com recursos subutilizados. Entre os fatores que ajudaram a virtualização tornar-se uma realidade, está a promessa de melhor utilizar recursos computacionais que de outra forma estavam sendo subutilizados dentro dos *Data Centers*.

Apesar da virtualização não ser um conceito novo, a virtualização na plataforma x86 vem se tornando mais popular nos últimos anos. A virtualização da plataforma x86 é mais complexa e de acordo com o teorema de Popek e Goldberg [2] não é uma plataforma que suporta de forma nativa a virtualização. Para que esse problema pudesse ser contornado, algumas propostas bastante criativas (como a Paravirtualização) foram concebidas e mostraram-se muito eficientes. Diversas propostas de virtualização para a plataforma x86 podem ser encontradas na Internet e na literatura, algumas já consolidadas como o VMWare e Xen e outras ainda em processo de consolidação, como o KVM.

A virtualização tem se mostrado uma tecnologia com grande aceitação no mundo corporativo. Praticamente os maiores fornecedores de TI contam com alguma solução de virtualização em seus portfólios. Vários casos de sucesso na consolidação de *Data Centers* podem ser encontrados na Internet validando os benefícios que a virtualização é capaz de trazer.

Para que a virtualização seja viabilizada, foram definidas algumas características que um ambiente virtualizado deve contemplar. Essas regras garantem que os recursos sejam divididos entre as MVs sem que uma sofra influência da outra. Tais características são as seguintes:

- **Isolamento**: Uma MV não pode influenciar no desempenho de outra MV;
- **Compatibilidade**: A MV deve ser completamente compatível com os padrões da plataforma que ela virtualizará (no caso do Xen, i386);
- **Encapsulamento**: Essa característica é o que torna a MV portável, pois viabiliza a cópia ou a movimentação de uma MV que está em execução de uma Máquina Física para outra;
- **Independência de Hardware**: As MVs devem ser independentes do Hardware em que elas estão sendo executadas. Por exemplo, é possível configurar interfaces de rede e outros dispositivos na MV que são completamente diferentes dos dispositivos físicos disponíveis.

Essas características tornam as máquinas virtuais um grande atrativo para a elaboração de um ambiente de testes. O custo, em geral, de manter um ambiente virtualizado é menor quando comparado com um ambiente tradicional de TI [3]. Entretanto, um dos maiores problemas ao se tratar de virtualização ainda é a perda de desempenho que esta tecnologia impõe [4][5].

A sobrecarga imposta por um ambiente virtualizado vem sendo tratada em diversas pesquisas [6][7][8]. Apesar do fator desempenho ser determinante em alguns ambientes de produção, em ambientes de testes e homologação pode ser algo passível de se conviver, pois muitas vezes o desempenho de um teste não é o principal objetivo. Apesar disso, é importante que o executor do teste tenha plena consciência de que, dependendo da técnica de virtualização utilizada para a construção do ambiente de testes, alguns fatores devem ser levados em conta, principalmente quando a técnica utilizada é a paravirtualização, que tem como principal característica a alteração do Sistema Operacional convidado.

Neste trabalho, serão mostradas algumas considerações em relação ao tipo de virtualização empregada e como estas podem influenciar no resultado final de um teste. Foram realizados testes de compilação em máquinas virtuais utilizando a Paravirtualização e a Virtualização Completa para verificar a

sobrecarga imposta por cada uma destas técnicas e por fim uma verificação das estruturas implementadas na paravirtualização e como estas influenciam os testes executados.

2. TRABALHOS RELACIONADOS

A virtualização é uma ferramenta bastante utilizada em simulação de ambientes reais. Quando abordado o uso de Máquinas Virtuais para a realização de testes de software, existem mais trabalhos voltados a máquinas virtuais de Linguagem (como JVM).

O uso de máquinas virtuais como simuladores se tornou mais difundido, pois a tecnologia de virtualização é mais comum em ambientes de infraestrutura como Sistemas Operacionais e dispositivos de rede (como *swithches* e roteadores). Trabalhos como [9] tratam da virtualização como ferramenta para simular um ambiente de rede completo. Segundo os autores, o uso de máquinas virtuais como simuladores de um ambiente real, tem como principal vantagem o complemento de técnicas mais formais como modelos matemáticos e modelos menos formais, como aqueles baseados em métodos empíricos.

Outro beneficio apontado pelos autores é o teste em paralelo de diversos sistemas com parâmetros diferentes, o que torna o processo como um todo mais ágil. A simulação de falhas de infra-estrutura é outro fator apontado como um dos grandes benefícios de se utilizar virtualização, como a falhas de rede, falhas de entrega de pacotes TCP/IP, entre outros, que tornam os testes mais próximos da realidade e, por conseqüência, mais completos.

Outro trabalho que tem como foco a simulação de um ambiente de infra-estrutura pode ser encontrado em [10]. Nesse artigo, os autores têm como foco os sistemas distribuídos. Em ambientes distribuídos, a simulação utilizando máquinas virtuais é ainda mais atraente, pois é possível avaliar todo o sistema nas condições mais adversas possíveis, como falha de um ou vários nós, criar condições de sobrecarga na rede virtual e entre outros testes que seriam consideravelmente limitados ou irreais, sem a ajuda da virtualização.

É importante notar que, em geral, o uso da virtualização ainda é um artifício complementar aos modelos mais formais de teste e simulação. Isto se deve em grande parte a sobrecarga da própria virtualização, que muitas vezes pode comprometer os resultados finais dos testes. Tal sobrecarga também é um assunto bastante explorado no meio acadêmico como pode ser notado nos trabalhos [11][12][13].

3. TECNOLOGIAS DE VIRTUALIZAÇÃO

Apesar da virtualização ser uma tecnologia concebida na década de 70, a virtualização na plataforma x86 ainda é um conceito novo. O ressurgimento da virtualização, principalmente no ambiente x86, deve-se principalmente à necessidade de melhor utilizar os recursos computacionais nestes equipamentos. Com o advento de processadores *multi-cores* e barateamento de memória física, os PC's convencionais, que compõe grande parte do parque do ambiente de TI das empresas, passaram a apresentar quantidades consideráveis de recurso ocioso.

O compartilhamento desses recursos foi uma das saídas para contornar o problema de ociosidade e a virtualização, conceito da década de 70, mostrou-se uma excelente idéia para concretizar a necessidade de compartilhamento de recursos. Entretanto, a plataforma x86 não foi concebida com o intuito de ser uma plataforma apta a ser virtualizada (como no caso de mainframes).

O principal problema da plataforma x86 foi demonstrado pelo teorema de Popek e Goldberg [2]. Esse teorema diz que uma plataforma pode ser virtualizada se, e somente se, todas as instruções sensíveis forem instruções privilegiadas. Simplificando, se uma instrução sensível for executada sem o privilégio correspondente, algum tipo de interrupção (*trap*) deve ocorrer. Na plataforma x86 isso não acontece e por esta razão, houve a necessidade de se criar artifícios para contornar o problema. A figura 1 ilustra o teorema de Popek e Goldberg.

As principais técnicas para contornar este problema são a Paravirtualização e a Tradução Binária que serão abordadas com mais detalhes a seguir.

3.1 Paravirtualização

De forma simplificada, a paravirtualização é assim chamada, pois a MV tem consciência de que está sendo executada em ambiente virtualizado. Isso só é possível devido a alterações realizadas no Kernel do Sistema Convidado. Essas alterações forçam o sistema convidado a utilizar estruturas de comunicação entre a máquina virtual e o monitor de máquinas virtuais.

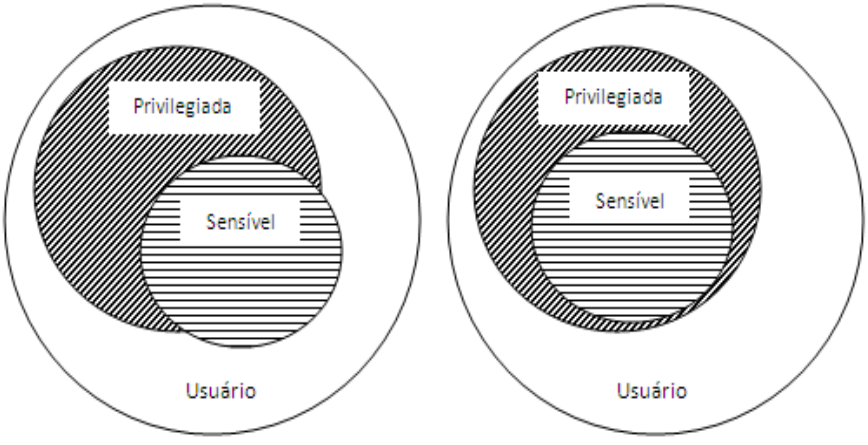

Figura 1. Teorema de Popek e Goldberg (a) Não Satisfaz a condição e (b) Satisfaz a condição [16].

O maior representante desta técnica de virtualização é o Xen [14]. Com o Xen os *drivers de* dispositivos que efetivamente acessam o hardware ficam no Domínio 0 (Dom0) que é uma espécie de máquina virtual privilegiada. As máquinas virtuais convidadas (DomU) tem seu código fonte alterado de tal forma que, ao executarem uma chamada de sistema que seja privilegiada (como atualização da tabela de páginas), é passada ao Monitor de Máquina Virtual que realiza todos os passos necessários. Estas chamadas de sistemas alterados recebem o nome de *Hypercalls* (em referencia ao *Hypervisor*).

No Xen, é utilizada uma estrutura chamada de I/O Ring. Essa estrutura tem a função de criar um mecanismo de I/O com maior desempenho em ambientes virtualizados. Com a implementação dessa estrutura, o Xen chega muito próximo ao desempenho de uma máquina tradicional com o Linux Nativo. A figura 2 mostra o funcionamento do I/O Ring.

O maior problema dessa abordagem é a necessidade de se alterar o código fonte do Sistema Operacional convidado (que nem sempre é aberto). Para portar o Linux foi necessário alterar 2995 linhas de código (algo em torno de 1,36% do total do código fonte). Já para o Windows XP foi preciso alterar 4620 linhas do código (algo em torno de 0,04% do total do código fonte) [14].

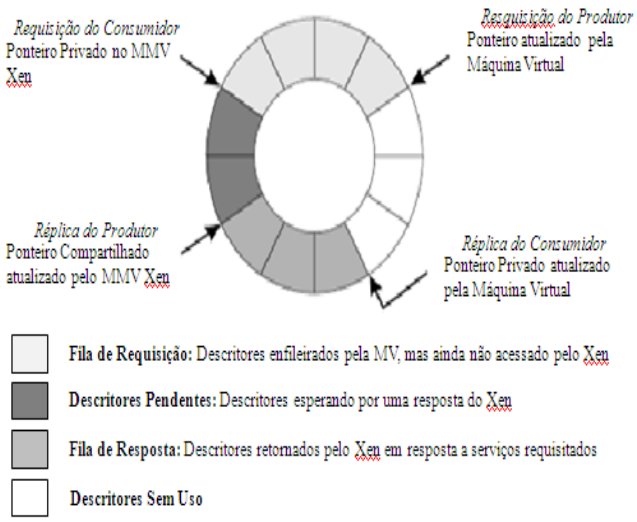

Figura 2. Estrutura de I/O Ring implementado no Xen [14].

Apesar do Xen ser o exemplo de maior representatividade da paravirtualização, também é possível utilizar a virtualização completa com o Xen. Mas para isso é necessário que o processador possua suporte à virtualização [15]. No presente trabalho o Xen será usado como monitor de máquinas virtuais e tanto a paravirtualização como a virtualização completa implementada pelo Xen serão usadas como cenários de teste.

3.2 Virtualização Completa

A Virtualização Completa é o oposto da paravirtualização e nela, o sistema operacional convidado não sofre nenhuma alteração em seu código fonte. Esta forma de virtualização costuma criar uma abstração da BIOS e o sistema operacional, por sua vez, é instalado logo acima dessa abstração.

Existem, basicamente, duas formas de virtualização completa. A primeira, que consiste em Tradução Binária (técnica amplamente usada pelo VMWare) e a virtualização através de emulação de dispositivos de hardware combinada com a tecnologia de virtualização no processador.

3.2.1 Tradução Binária

Esta técnica consiste em verificar todas as instruções antes de serem executadas. No caso de alguma instrução sensível, porém não privilegiada, ser executada, é inserido um ponto de parada nesta instrução. Esses pontos costumam ser trocas de contextos para o Monitor de Máquinas Virtuais (MMV) e após a execução da instrução Na arquitetura IA-32 existem ao todo dezessete instruções consideradas sensíveis e não privilegiadas[15]. De forma geral, essas instruções podem ser enquadradas em duas categorias:

- **Proteção a Referências de Sistema**: Tais instruções se referem à proteção de armazenamento, de memória ou alocação de endereço. Um exemplo deste tipo de instrução é o MOVE, que move o valor de um registrador de propósito genérico para o registrador CS.
- **Instruções Sensíveis de Registrador:** Este conjunto de instruções se refere a leituras ou alterações em registradores relacionados aos recursos da máquina e endereços de memória. Instruções que estão nesta categoria são registradores de interrupção ou o registrador do *clock.*

A técnica de Tradução Binária acrescenta uma complexidade considerável ao MMV, porém consegue prover um conjunto bastante completo de instruções da arquitetura IA-32 que possibilita a utilização de Sistemas Operacionais sem alteração de código.
O maior exemplo de monitor de máquina virtual que utiliza esta técnica é o VMWare. A utilização da tradução binária em conjunto com as outras técnicas de virtualização de I/O e memória, faz com que o VMWare consiga atingir um desempenho muito próximo ao desempenho de um ambiente tradicional sem virtualização.

3.2.2 Emulação e Virtualização no Processador

A emulação, na verdade, não é virtualização [16]. Frequentemente ocorre este tipo de engano, pois alguns monitores de máquinas virtuais utilizam esta técnica para disponibilizar alguns dispositivos de I/O (como interface de rede e disco) para as máquinas virtuais. A técnica de emulação de alguns dispositivos é utilizada no KVM [17] e no Xen.
Esta forma de disponibilizar um dispositivo emulado para o sistema operacional tem como principal objetivo evitar a alteração do código fonte do Kernel. Entretanto, a emulação adiciona grande sobrecarga sobre o sistema [18], pois o comportamento do dispositivo emulado deve ser simulado pelo processador.
Já a virtualização no processador é uma técnica proposta pelos fornecedores de processador como a AMD e Intel. Um processador convencional, em geral, possui dois modos de execução, o modo *root* (privilegiado) e modo *non-root* (não-privilegiado). Os sistemas operacionais atuais foram projetados para executar em modo privilegiado e não funcionam corretamente se estiverem sendo executados em qualquer modo que não o privilegiado.
Para contornar este problema e facilitar a implementação de monitores de máquinas virtuais mais eficientes, os fornecedores

criaram um terceiro estado de execução, chamado de modo VMX. Neste estado de execução, a máquina virtual acredita estar em modo privilegiado, mas quando uma instrução privilegiada é executada, o processador realiza uma troca de contexto para o modo privilegiado propriamente dito e após a execução da instrução volta para o modo VMX.

4. METODOLOGIA

Para verificar a sobrecarga das duas técnicas de virtualização abordadas neste trabalho, será executado um *benchmark* de compilação do Kernel do Linux. Como referência, o teste foi executado no Linux convencional com as mesmas configurações das Máquinas Virtuais (as configurações podem ser verificadas na Tabela 1). Com isto, o objetivo é identificar a sobrecarga dos dois métodos de virtualização em comparação a um ambiente convencional.

Para cada configuração, foi executado um total de 15 compilações do Kernel. Após a execução, foram calculados a média, o desvio padrão e a distribuição *stundent-t* com 95% de confidencialidade [19]. A distribuição *student-t* indica que 95% dos resultados estarão entre a média e o desvio calculado pela distribuição. A grande vantagem ao utilizar-se este método é a pequena quantidade de dados necessários para obter o resultado (diferente do que acontece com a distribuição normal).

Tabela 1. Configuração das Máquinas para a realização dos Testes

Técnica Utilizada	Hardware	Software
Linux Tradicional	1xAthlon64 Dual Core Processor[1], 2GB de memória.	Fedora Core 8, Kernel 2.6.21.7-5 SMP.
Linux Paravirtualizado	2 VCPUs, 2GB de memória.	Fedora Core 8, Kernel 2.6.21.7-5.fc8xen, Xen 3.1.2
Linux Virtualização Completa	2 VCPUs, 2GB de memória.	Fedora Core 8, Kernel 2.6.21.7-5 SMP.

A compilação do Kernel do Linux (versão 2.6.18)foi escolhida, pois é um *benchmark* amplamente utilizado e aceito como parâmetro de comparação no meio acadêmico. A compilação do Kernel possui diversas características como operações intensivas de I/O e grande utilização de memória cache. Entretanto, sua maior característica é a considerável utilização do processador. Este *benchmark* pode ser influenciado por diversos fatores como vazão de disco, capacidade do sistema de paralelizar tarefas, entre outros. Por esta razão a compilação do Kernel é um *benchmark* bastante completo, que tem condições de avaliar o sistema como um todo.

5. RESULTADOS

Após a execução dos testes, foi observado que a técnica de paravirtualização consegue se aproximar bastante do desempenho de uma máquina real com o Linux convencional. Isto se deve à otimização das estruturas de I/O alteradas no Kernel. Com essas alterações não há necessidade de emular nenhum dispositivo.

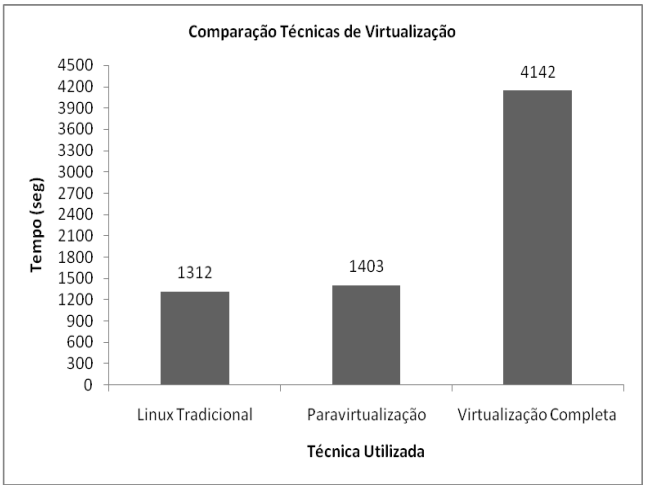

Figura 3. Resultados obtidos em cada uma das técnicas de virtualização.

Na técnica de virtualização completa, ainda é observada uma sobrecarga muito grande por conta da emulação. Apesar do teste de compilação do Kernel ser um *benchmark*, que possui característica de uso intensivo do processador, a sua dependência de velocidade de I/O é determinante no resultado final. Por conta da dependência de I/O essa técnica de virtualização ainda sofre de grande sobrecarga quando o quesito é desempenho. Na figura 3, é apresentado o gráfico dos resultados com a compilação do Kernel.

Outro dado importante é a variação dos resultados com a virtualização completa. Em um ambiente sem virtualização, os resultados sofreram pouca variação. O mesmo comportamento pode ser observado com a paravirtualização (observa-se uma variação maior, entretanto menos acentuada). Quando analisados os resultados da virtualização completa, pode-se verificar que eles sofreram grandes variações. Isto se deve em grande parte à demora de resposta dos dispositivos emulados. Ao emular um dispositivo, este torna-se muito dependente do poder de processamento do hardware real. Como o processador estava sendo disputado entre a compilação do Kernel, a emulação dos dispositivos e outras tarefas (como execução de algoritmos de paginação do próprio kernel da máquina hospedeira e da máquina virtual) a variação tende a ser maior.

Na Tabela 2, são apresentados os resultados da execução do *benchmark* (segunda coluna), o desvio padrão (terceira coluna) e a distribuição *student*-t (quarta coluna) com confiabilidade de 95%. É interessante observar que ao utilizar a virtualização completa os resultados tiveram maior oscilação se comparados com a paravirtualização e o ambiente real. A oscilação se deve principalmente à sobrecarga causada pela emulação de alguns dispositivos de I/O e as trocas de contexto entre modo real e virtual do processador.

6. CONCLUSÃO

A virtualização é sem dúvida uma grande ferramenta para execução de testes e simulação. As características de um ambiente virtualizado são muito atrativas quando o objetivo é a criação de

[1] Um processador com 2 Cores, portanto 2 processadores.

um ambiente de testes mais próximo do real. A virtualização não deve ser encarada como uma substituta de modelos formais para a execução de testes, mas sim como um complemento. Por meio da virtualização, é possível executar e analisar testes que de outra forma seriam difíceis e muitas vezes inviáveis.

Apesar de todos estes benefícios da virtualização, é importante que os executores do teste tenham consciência de que há ainda o problema de desempenho que estes ambientes oferecem. Ainda não há nenhum método formal para analisar a sobrecarga causada pelos monitores de máquinas virtuais. Dessa forma para testes que são muito dependentes de desempenho (como sistemas de tempo real) uma boa opção seria o uso de máquinas virtuais paravirtualizadas. Em contrapartida, em testes que são dependentes do código do sistema operacional (como *drivers* de dispositivos), a virtualização completa mostra-se atrativa, pois neste modo de virtualização o código fonte do Sistema Operacional não sofre alterações.

Outro beneficio da virtualização é o aumento da flexibilidade dos ambientes de testes. É possível instalar MVs utilizando uma grande variedade de configurações de *hardware* (como memória, processador, interfaces de rede, etc). Em geral, esse tipo de configuração não é um obstáculo para o executor dos testes, pois os MMVs possuem ferramentas que possibilitem a alteração dessa configuração de forma intuitiva (através de interfaces gráficas). No Xen, a memória e a prioridade da MV podem ser alteradas sem haver necessidade de reiniciar ou reinstalar a MV, o que torna essa tecnologia mais atraente para ser utilizada em um ambiente de testes.

Tabela 2. Resultados dos Testes e Variação

Técnica Utilizada	Desempenho (seg)	Desv. Padrão	Distribuição Student-t (95%)
Linux Tradicional	1312,33	7,44	3,39
Linux Paravirtualizado	1402,53	30,55	13,90
Linux Virtualização Completa	4142,13	783,06	356,38

7. REFERÊNCIAS

[1] Goldberg, R. 1974 Survey of virtual machine research. In Survey of Virtual Machine Research. IEEE Computer. pp 34-45.

[2] Popek, G. J. e Goldberg, Robert 1974. Formal Requirements for Virtualizable Third-Generation Architectures. Comm. ACM, pp. 412-421.

[3] Whitaker, A., Cox, R. S., Shaw, M., Gribble, S. D. 2005 Rethinking the Design of Virtual Machine Monitors. IEEE Computer. pp 57-62.

[4] L. Cherkasova, D. Gupta, and A. Vahdat. 2007 When virtual is harder than real: Resource allocation challenges in virtual machine based IT environments. Technical Report HPL-2007-25.

[5] E. Bolker and Y. Ding. 2005 Virtual performance won't do: Capacity planning for virtual systems. Proc. 31th Int.Computer Measurement Group Conf, 1:39.

[6] Zhang, X., Dong, Y. 2008 Optimizing Xen VMM Based on Intel Virtualization Technology. Proceedings of the 2008 International Conference on Internet Computing in Science and Engineering. pp. 367-374.

[7] I. Ahmad, J. M. Anderson, A. M. Holler, R. Kambo, and V. Makhija, 2003 An Analysis of Disk Performance in VMware ESX Server Virtual Machines in Workload Characterization, 2003. WWC-6. 2003 IEEE Internatinal Workshop, pp. 65–76.

[8] Nakajima, J., Mallick, A. K. 2007 Hybrid-Virtualization – Enhanced Virtualization for Linux, Proceedings of The Linux Symposium.

[9] Wang, R. T., Browne, J. C. 1981 Virtual machine-based simulation of distributed computing and network computing in Proceedings of the 1981 ACM SIGMETRICS conference on Measurement and modeling of computer systems. pp. 154-156.

[10] Hérault, T., Largillier, T., Peyronnet, S., Quétier, B., Cappello, F., Juan, M. 2009 High accuracy failure injection in parallel and distributed systems using virtualization in Proceedings of the 6th ACM conference on Computing frontiers. pp. 193-196.

[11] VMware, 2007 A Performance Comparison of Hypervisors, http://www.vmware.com/pdf/hypervisor_performance.pdf.

[12] Che, J., He, Q., Gao, Q., Huang, D., 2008 Performance Measuring and Comparing of Virtual Machine Monitors in Proceedings of the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing. pp. 381-386.

[13] XenSource, 2007 A Performance Comparison of Commercial Hypervisors, http://www.xensource.com/Documents/hypervisor_performance_comparison_1_0_5_with_esx-data.pdf.

[14] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, 2003 Xen and the Art of Virtualization, ACM Symposium on Operating Systems Principles (SOSP).

[15] Intel Virtualization, 2005 Technology Specification for the IA--32 Intel Architecture.

[16] J. E. Smith, R. Nair 2005 Virtual Machines – Versatile Platforms for System and Process, 1ª ed., vol. 1. Morgan Kaufmann.

[17] Qumranet, 2006. Kvm: Kernel-based virtualization driver. http://kvm.qumranet.com/kvmwiki/Documents.

[18] J. E. Smith, R. Nair 2005 The Architecture of Virtual Machines in IEEE Computer Society Press. pp. 32-38.

[19] On Student's 1908 Article "The Probable Error of a Mean", Journal of the American Statistical Association.