Avaliação de desempenho entre KVM e VirtualBox em um ambiente de Desktop-as-a-Service

Demis Gomes[1], Nichene Verçosa[1], Daniel Carvalho[1], Victor Medeiros[1], Glauco Gonçalves[1]

[1]Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco (UFRPE)
CEP 52.171-900 – Recife – PE – Brasil

{demismg72,nichenevercosa,danieljosecarvalho}@gmail.com

{victor,glauco.goncalves}@deinfo.ufrpe.br

Abstract. *Desktop-as-a-Service is a Cloud Computing model that provides a user with access to their desktop as a virtualized service in Cloud, offering scalability, fault tolerance, availability and cost reduction. To offer better performance in this environment, it is necessary to analyze alternative hypervisors, checking which of them consume less resources from a given load on the system. This article compares performance between the KVM and VirtualBox hypervisors, specifically their consumption of CPU and memory. In our experiments, KVM was 52% more efficient than VirtualBox regarding to CPU consumption and achieved similar results related to memory consumption.*

Resumo. *Desktop-as-a-Service é um dos modelos da Computação em Nuvem que provê o acesso de um usuário a seu desktop como um serviço virtualizado na Nuvem, oferecendo escalabilidade, tolerância à falhas, disponibilidade e redução de custos. Para oferecer um melhor desempenho neste ambiente, é necessário analisar alternativas de hypervisors, verificando quais delas consomem menos recursos a partir de uma determinada carga no sistema. Este artigo compara o desempenho entre os hypervisors KVM e VirtualBox em relação ao consumo de CPU e de memória. Em nossos experimentos, verificou-se que o KVM foi 52% mais eficiente do que o VirtualBox em relação ao consumo de CPU e obteve resultados semelhantes referentes ao consumo de memória.*

1. Introdução

Computadores convencionais ainda são a escolha mais comum em redes locais de pequeno e médio porte, como laboratórios de informática, departamentos, pequenas empresas e *callcenters*. Nestes ambientes, as atividades realizadas pelos usuários são homogêneas, isto é, os usuários executam aplicações semelhantes na maior parte do tempo em que utilizam seu computador. Esses ambientes são adequados ao uso de soluções de Desktop-as-a-Service (DaaS).

DaaS preconiza a execução da área de trabalho, com aplicações e arquivos do usuário em servidores virtualizados na Nuvem. A área de trabalho é apenas mostrada ao usuário na tela de seu monitor. O controle da área de trabalho é feito pelos dispositivos de entrada, como mouse e teclado. Com o emprego deste conceito, os computadores têm a única função de acessar o servidor, fazendo com que haja a possibilidade de

adquirir máquinas mais leves e que consomem menos energia elétrica. Desta forma, é possível reduzir os custos de toda a infraestrutura, além de prover benefícios existentes na Computação em Nuvem como escalabilidade, disponibilidade, tolerância à falhas e segurança. Uma opção de terminal são os *thin clients*[Deboosere et al. 2012], dispositivos que têm a função de apenas acessar o servidor remoto, que podem reduzir o consumo de energia elétrica em cerca de 80% em relação aos computadores convencionais [Dasilva et al. 2012].

Para prover uma solução de DaaS eficaz, faz-se necessário verificar quais os fatores mais importantes para a criação desta plataforma, os softwares mais adequados para a realização desta tarefa e os possíveis gargalos de desempenho que podem ocorrer. Uma fonte de gargalo é a camada de virtualização, chamada *hypervisor*, que gerencia o acesso dos recursos da máquina física aos servidores hospedados em máquinas virtuais (VMs, *Virtual Machines*). Um *hypervisor* que gere alta carga no sistema pode influenciar diretamente no desempenho de toda a solução, uma vez que geralmente há vários servidores sobre o *hypervisor*. Deste modo, o objetivo do artigo é comparar dois tipos de *hypervisor*s open-source - VirtualBox e KVM - para verificar qual deles é mais indicado para ambientes DaaS.

O artigo está dividido em seis seções. A segunda seção discorre os trabalhos relacionados que contribuíram para a motivação dos objetivos deste artigo. A seção 3 apresenta uma visão geral de DaaS, discutindo sobre uma plataforma DaaS proposta por nós explorando suas características e funcionalidades. A seção 4 apresenta o experimento de avaliação feito com parte da plataforma já utilizável, comparando o desempenho dos *hypervisor* KVM e VirtualBox. Nas últimas duas seções, concluímos o artigo e propomos futuras pesquisas.

2. Trabalhos Relacionados

Alguns trabalhos na literatura buscaram comparar *hypervisors* por diferentes fatores, como custos e desempenho. Outros formularam técnicas de avaliação automáticas para simular o comportamento de usuários no sistema a fim de verificar a qualidade de experiência provida pelo sistema.

Chang, et al[Chang et al. 2013] concentraram-se na análise de custos, comparando cinco *hypervisors* (VMware ESX, Microsoft Hyper-V, Proxmox VE, Ubuntu KVM e XenDesktop) em relação a custo por desempenho e ROI (*Return On Investment*). Após as comparações, os resultados demonstraram que o ESX possui o melhor ROI inicial, mas para cargas de trabalho maiores o Proxmox VE oferece um resultado melhor. Em referência ao custo por desempenho, o Proxmox VE obteve os melhores resultados. Os autores concluíram que o Proxmox seria a melhor solução devido ao menor valor de custo por desempenho e por oferecer um bom ROI com altas cargas de trabalho durante um período de 5 anos.

Opsahl [Opsahl 2013], também comparou *hypervisors*, porém todos open-source. As três opções foram KVM, XenDesktop e VirtualBox. Ele comparou as opções utilizando benchmarks para medir a velocidade de leitura e escrita de dados na memória principal e na cache, do desempenho da CPU e da taxa de I/O de disco. Os resultados demonstraram que o KVM obteve desempenho melhor na leitura e escrita de memória, no desempenho da CPU e levemente inferior ao Xen nos testes com I/O, sendo o VirtualBox

o *hypervisor* com menor desempenho. Opsahl concluiu que o KVM ultrapassou o Xen no período entre a leitura dos artigos base para seu trabalho e os testes realizados.

Ferramentas que emulavam o comportamento de usuários no sistema também foram criadas. Shabaitah [Shabaitah 2014] avaliou o desempenho de sistemas DaaS baseado em plataforma proprietária (Windows) por meio de um método automático para emulação do comportamento de usuários. Neste trabalho, buscou-se emular três tipos de usuários: leve, pesado e multimídia. O primeiro digitava lentamente e abria programas leves, como o Word. O segundo utilizava mais recursos, abrindo vários programas de uma vez como o browser, o Excel e o PowerPoint. O último executava um vídeo e uma música curta. Com base nestes usuários, Shabaitah buscou entender o comportamento de cada grupo de usuários no sistema, para que desta forma fosse oferecida a quantidade de recursos mais eficiente para cada um deles.

Nosso experimento buscou comparar os *hypervisors* VirtualBox e KVM, a fim de verificar se os resultados assemelhavam-se aos de [Opsahl 2013] em um sistema DaaS. Para isso, criamos uma ferramenta de avaliação automática seguindo o trabalho de [Shabaitah 2014], com a diferença que automatizamos o comportamento do usuário em um sistema operacional Linux. Desta forma, iremos verificar a carga no sistema e comparar o desempenho dos hypervisors open-source para concluir qual poderia ser uma melhor opção para um ambiente DaaS.

3. Uma visão geral de Desktop-as-a-Service

DaaS pode ser definido como uma das categorias de serviço da Computação em Nuvem que se caracteriza pela execução da área de trabalho (*desktop*) e aplicações do usuário remotamente, em máquinas virtuais na Nuvem. Nesta Nuvem, a área de trabalho, aplicações e arquivos do usuário são hospedados, permitindo que ele acesse sua área de trabalho a partir de terminais, ou seja, dispositivos como *tablets*, *smartphones*, *thin clients*, computadores convencionais ou *notebooks* que possuam acesso à Nuvem através da Internet ou da rede local. A Figura 1 apresenta a arquitetura em camadas proposta em nossa plataforma DaaS. Outras plataformas apresentam estrutura semelhante com algumas particularidades.

Figura 1. Arquitetura da plataforma

O terminal, também chamado de cliente, é um dispositivo que acessa o servidor e mostra ao usuário apenas a tela de sua área de trabalho. Sua função é de enviar ao servidor, por meio dos protocolos, os comandos referentes às ações do usuário, que são recebidas através da interação deste com os dispositivos de entrada como mouse e teclado.

Em termos de serviço, são propostos três. O Serviço de Desktop Virtual transmite a tela remotamente do servidor ao terminal, através de protocolos de acesso específicos. Por sua vez, o Serviço de Administração consiste na gerência de usuários centralizada e caracterizada pelo controle de usuários, correspondendo à criação, edição e remoção de usuários de maneira simples. Além disso, o administrador pode atribuir uma política de uso do sistema que inclui permissões específicas para grupos de usuários. Por fim, o Serviço de Boot Remoto compreende: a entrega do IP ao terminal por meio do DHCP(*Dynamic Host Configuration Protocol*); o boot remoto que geralmente ocorre através do protocolo PXE (*Preboot Execution Environment*); e a escolha de qual dispositivo os recursos serão utilizados, ou seja, se o terminal utilizará seus recursos ou apenas acessará o servidor.

Com relação ao gerenciamento de nuvem, destacam-se as funções de balanceamento de carga, migração e eficiência de consumo de energia. Em uma plataforma de consumo eficiente de recursos, é necessário a utilização de algoritmos para estas funções, a fim de manter um bom desempenho para os usuários e utilizar a capacidade da máquina física de maneira mais eficiente. O balanceamento de carga permite que os servidores de aplicação sejam distribuídos pela nuvem, oferecendo melhor desempenho ao usuário.

A virtualização auxilia a escalabilidade, a administração e o backup dos dados, uma vez que máquinas virtuais de diferentes configurações são criadas rapidamente e podem ter seu estado salvo a qualquer momento. A flexibilidade proporcionada pela virtualização também permite que usuários acessem diferentes tipos de sistemas operacionais em diferentes configurações de RAM, CPU e capacidade de HD. A gerência das máquinas virtuais é feita pelo *hypervisor*, camada de software que emula o hardware da máquina física para as máquinas virtuais. O *hypervisor* permite ligar, desligar, salvar o estado das máquinas e, a maioria deles, realizar migração de VMs entre servidores.

A plataforma deve prover um sistema de armazenamento das imagens das máquinas virtuais e dos dados dos usuários. É importante manter um backup dos dados, que pode ser em outro sistema externo à plataforma ou interno, provendo redundância de dados caso um servidor falhe, impedindo que o usuário tenha seus arquivos e aplicações perdidas. Na base da estrutura existe uma camada física que corresponde aos servidores e dispositivos de rede, onde são hospedados todos os serviços. Os servidores físicos são conectados por switches e roteadores, e nestes servidores são executados programas de gerenciamento e compartilhamento de recursos.

4. Experimentos

4.1. Metodologia

Os experimentos focam na análise da camada de virtualização por meio da comparação do desempenho do VirtualBox e do KVM para a hospedagem dos serviços de boot remoto e do servidor de aplicações de um DaaS quando submetidos a um pequeno conjunto de usuários. Assim, pretende-se indicar qual *hypervisor* seria mais indicado para oferecer estes serviços numa plataforma DaaS.

Para a realização do experimento, desenvolvemos uma ferramenta de geração automática de carga que emula o comportamento dos clientes e coleta resultados referentes ao uso de recursos do sistema em termos de CPU e memória usada. A quantidade de

clientes, bem como o número de amostras (repetições) do experimento, são parâmetros da ferramenta. A cada amostra, a ferramenta executa o experimento que compreende dos seguintes passos: lançamento de VMs que emulam os terminais dos usuários; execução do boot remoto por meio do protocolo PXE, padrão na indústria; autenticação e carregamento da área de trabalho do usuário do terminal; execução das tarefas associadas ao perfil do usuário e coleta das métricas.

As atividades do usuário na ferramenta seguem o trabalho de [Shabaitah 2014], no qual eram definidas aplicações a serem executadas e ações que o cliente faria. O cliente definido na ferramenta executa as seguintes tarefas: acessa um site no Firefox, abre um arquivo de 1,2 MB no LibreOffice Writer e edita um parágrafo de três linhas, abre uma planilha de 256 KB e edita catorze células, abre uma apresentação de 700 KB, acessa outro site, executa o gcalctool e abre um documento PDF de 3,2 MB. As atividades foram automatizadas pelo *xdotool*[1] e a carga gerada no servidor é medida pelo *dstat*[2] e coletada para análise. A ferramenta também pode ser configurada para outros perfis, sendo necessário apenas adicionar um novo perfil e as atividades que ele fará durante a sua execução.

O LTSP Cluster[3] foi utilizado para prover o serviço de DaaS. O experimento foi realizado utilizando dois servidores: Root Server, que tem a função de ser o servidor de boot remoto, servidor de autenticação e de arquivos, e o Application Server, que executa o serviço de desktop virtual. A infraestrutura de testes utiliza um computador para hospedagem de servidores virtuais com processador Intel Core i7 2.93 GHz 8 cores, 8GB de RAM, e um computador para hospedagem de clientes para testes de carga com Intel Quadcore 2.66 GHz, 4GB de RAM. A Figura 2 mostra o esquema do experimento. Neste experimento, buscamos medir o consumo de memória e de CPU dos servidores que hospedam o serviço ao longo do experimento.

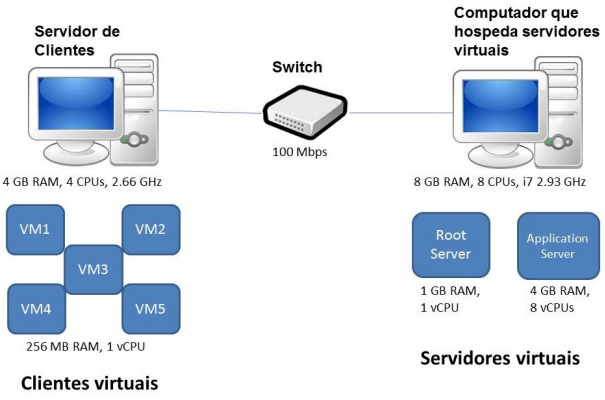


Figura 2. Esquema do experimento de comparação de hypervisors

O servidor de clientes executa a ferramenta de avaliação automática que controla o experimento e inicia os clientes hospedados em máquinas virtuais por meio do *hypervisor* VirtualBox. A escolha pelo VirtualBox se deu após a verificação de que não havia maiores diferenças no uso de um ou outro *hypervisor* para a hospedagem dos clientes e

[1]http://www.semicomplete.com/projects/xdotool/
[2]http://dag.wiee.rs/home-made/dstat/
[3]https://www.ltsp-cluster.org/

por praticidade na automação do experimentos. Cada VM cliente foi configurada com 256 MB de memória RAM e uma CPU virtual. Com esta configuração esperamos emular a capacidade de hardware comum em *thin clients*. O outro computador hospeda os servidores virtuais Root e Application Server. O servidor Root é configurado com 1GB de RAM e 1 CPU virtual. O Application Server contém 4GB de RAM e 8 CPUs. Ambos os servidores têm como SO o Ubuntu 12.04. As configurações dos servidores virtuais foram ajustadas de modo a obter o melhor desempenho do sistema de DaaS, medido pelo tempo de conclusão das tarefas dos usuários, com a restrição dada pelo hardware do computador onde executam.

Para obtermos um tempo base de duração das atividades de cada cliente em um ambiente real, executamos as tarefas do cliente cinco vezes em um notebook convencional de 2GB de RAM, Intel Core TM 1.73 GHz com o Ubuntu 12.04 como sistema operacional. As tarefas foram executadas com um tempo médio de 31,20 segundos, com intervalo de confiança de 95% no valor de 1.12s.

4.2. Resultados

A Tabela 1 mostra a comparação do consumo de CPU no Application Server.

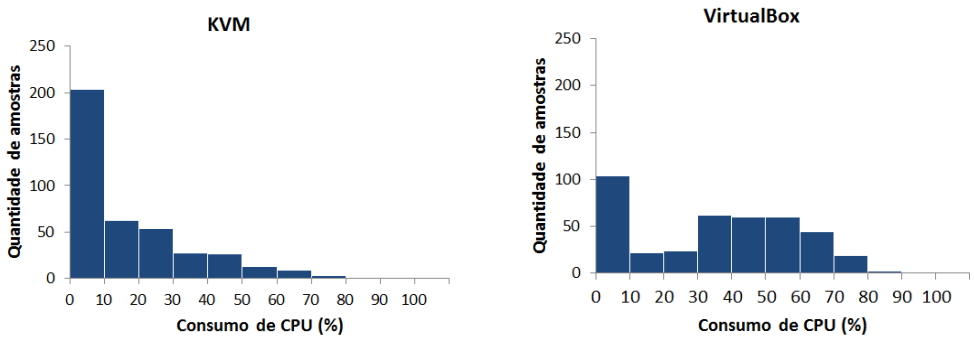
Tabela 1. Consumo de CPU do Application Server utilizando VirtualBox e KVM

Consumo de CPU	KVM	VirtualBox
Média de consumo (%)	16.46	34.46
Desvio padrão	17.46	24.39
Valor máximo (%)	73.65	82.75
Valor mínimo (%)	0	0

Verifica-se que o Application Server hospedado no KVM consome em média, menos CPU do que no VirtualBox. Além disso, o desvio padrão mostra que o resultado do VirtualBox é ainda mais variável do que o do KVM. Para confirmar o menor consumo de memória e a menor variabilidade do KVM, comparamos a partir de histogramas o consumo de CPU durante o experimento. A comparação é feita na Figura 3. Na Figura 3(a), vemos que a maioria dos valores do KVM situa-se abaixo dos 30% de consumo de CPU. Na Figura 3(b), percebemos que o VirtualBox possui muitos valores na faixa entre 30 e 70% de consumo de CPU.

A tabela 2 mostra os resultados em relação à memória. Percebemos que os valores são muito próximos. Assim, pode-se concluir, que em nosso experimento há vantagem do KVM em relação ao VirtualBox no quesito consumo de CPU, semelhante aos experimentos de Opsahl, que também indicava uma melhor performance do KVM neste quesito.

A Tabela 3 mostra o tempo em que os clientes realizaram suas atividades simultaneamente no sistema de uma das amostras. Estes valores são um indicativo da qualidade de experiência do usuário, e o impacto do consumo de CPU pode influenciar este quesito. A partir destes resultados, podemos verificar que os clientes que executam suas aplicações no Application Server que é executado no KVM obtiveram um desempenho melhor do que no VirtualBox. Neste cenário, percebemos que o tempo de execução dos clientes no KVM, em média, foi 58% maior do que o tempo base em um computador convencional (31,20 segundos), contra 166% de aumento do tempo de execução no Application Server



(a) (b)

Figura 3. Histogramas de comparação de hypervisors

Tabela 2. Comparação do consumo de memória do Application Server hospedado em diferentes hypervisors

Consumo de memória	KVM	VirtualBox
Média de consumo (%)	34.17155	35.6743
Desvio padrão	18.97558	19.16496
Valor máximo (%)	56.43	56.18
Valor mínimo (%)	8.48	9.42

no VirtualBox. O primeiro cliente obteve um tempo maior porque ele iniciou primeiro, e enquanto tentava abrir o primeiro arquivo, os outros quatro clientes iniciaram o boot de uma única vez. O momento do boot é onde há o maior consumo de disco e rede, e isso fez com que o tempo de abertura do arquivo fosse bem maior que os demais.

Nos clientes que executaram suas aplicações no Application Server sendo executado no VirtualBox, os tempos foram maiores e mais variáveis. Os últimos dois tempos, bem maiores, podem ser explicados pelo maior consumo de CPU do VirtualBox e pela execução simultânea de tarefas com maior consumo de recursos. Os três primeiros clientes iniciaram primeiro e, enquanto os outros dois ainda estavam iniciando, executaram praticamente todo o script. Quando os outros dois iniciaram, a carga de trabalho no Application Server era bem maior, levando os clientes 4 e 5 a executarem suas tarefas em um tempo maior que os demais.

5. Conclusão e Trabalhos Futuros

A partir deste artigo, pudemos verificar que os resultados de nossos experimentos foram semelhantes aos de [Opsahl 2013] para ambientes DaaS no consumo de CPU. Tal resultado pode ser explicado pela forma como o KVM é executado, uma vez que é embutido no kernel do Linux, diferentemente do VirtualBox que é um *hypervisor* executado sobre o kernel. Desta forma, o KVM acaba sendo uma opção mais eficiente devido ao melhor uso de recursos da CPU.

Os testes apresentados neste trabalho são iniciais dada a pequena escala. Assim, espera-se realizar experimentos de carga com um número maior de repetições e de clientes para verificarmos resultados referentes à carga gerada no Application Server a partir de uma alta demanda de clientes leves e pesados.

Tabela 3. Comparação do tempo de execução das tarefas por cliente em diferentes hypervisors

Valores analisados	KVM	VirtualBox
Cliente1	81.17s	51.05s
Cliente 2	43.59s	55.60s
Cliente 3	40.50s	51.72
Cliente 4	38.30s	113.56s
Cliente 5	41.38s	140.79s
Média dos tempos	48.99s	82.54s
Intervalo de Confiança(95%)	+/- 15,85s	+/- 36,72s

Outro trabalho futuro a ser conduzido é a avaliação do uso de containers para a Cloud Computing, uma vez que esta estratégia reduz o overhead de processamento existente em máquinas virtuais. Na comparação de desempenho entre containers Docker[4] e VMs usando o KVM [Seo et al. 2014], o Docker demonstrou ter vantagens em relação ao custo de armazenamento, o desempenho de processamento e o tempo de boot. Tal resultado decorre do maior consumo de recursos pela virtualização apenas para o SO convidado, limitando a quantidade de VMs geradas e aumentando o tempo de carregamento da VM. No Docker, o container é uma extensão do SO contendo que permite aproveitar memória e armazenamento do hospedeiro de modo mais eficiente. Desta forma, utilzar a virtualização por containers pode ser uma alterativa futura para ambientes DaaS.

6. Agradecimentos

Agradecemos à FACEPE pelo apoio na forma de Bolsa de Iniciação Científica através do projeto Desenvolvimento de Plataforma de DaaS para Laboratórios de Ensino de Informática

Referências

Chang, B. R., Tsai, H.-F., and Chen, C.-M. (2013). Empirical Analysis of Server Consolidation and Desktop Virtualization in Cloud Computing. *Mathematical Problems in Engineering*, 2013:1–11.

Dasilva, D.-A., Liu, L., Bessis, N., and Zhan, Y. (2012). Enabling Green IT through Building a Virtual Desktop Infrastructure. In *2012 Eighth International Conference on Semantics, Knowledge and Grids (SKG)*, pages 32–38.

Deboosere, L., Vankeirsbilck, B., Simoens, P., De Turck, F., Dhoedt, B., and Demeester, P. (2012). Cloud-Based Desktop Services for Thin Clients. *IEEE Internet Computing*, 16(6):60–67.

Opsahl, J. M. G. (2013). Open-source virtualization: Functionality and performance of Qemu/KVM, Xen, Libvirt and VirtualBox.

Seo, K.-T., Hwang, H.-S., Moon, I.-Y., Kwon, O.-Y., and Kim, B.-J. (2014). Performance comparison analysis of linux container and virtual machine for building cloud.

Shabaitah, A. R. (2014). Server-Based Desktop Virtualization.

[4]https://www.docker.com/