Comunicado Técnico 102
ISSN 1677-8464
Dezembro, 2010
Campinas, SP

Fotos: <http://www.google.com.br/imgres>

Virtualização com o *Xen*: instalando e configurando o ambiente

Leandro Carrijo Cintra

O documento descreve o processo de instalação de ambientes virtuais para servidores, utilizando o hypervisor *Xen* e tendo-se um sistema Linux Ubuntu (*Lucid release*) como *Dom0*. Descreve-se o processo de instalação do *Xen* 3.3 e 4.0 e também a instalação de hóspedes HVM (totalmente virtualizados) e PV (paravirtualizados). São apresentados também alguns testes comparando o desempenho de sistemas rodando em máquinas reais e em máquinas virtuais.

A virtualização

A virtualização possibilita que dois ou mais sistemas operacionais compartilhem os recursos de uma mesma máquina física, permitindo que os eles executem concomitantemente. Originou-se na década de 60 com a introdução da tecnologia de hypervisor em *mainframes* da IBM; atualmente, vem passando por um processo de reflorescimento e é apontada como uma das principais ferramentas para a administração eficiente de centros de processamento de dados (*data centers*). A virtualização promete racionalizar o uso de servidores nesses centros e também possui características que indicam que o seu uso trará benefícios para a área de Tecnologia da Informação (TI) das organizações em geral. Por isso, é importante que as organizações estejam atentas e acompanhem a evolução dessa área dentro da TI. Com esse intuito, o presente trabalho faz uma avaliação do *XEN*, uma interessante ferramenta "*open source*" para a virtualização de servidores.

Vantagens da virtualização

A virtualização traz uma série de benefícios e dentre os principais pode-se citar:

- **Consolidação de servidores**: essa certamente é uma das grandes vantagens oferecidas pela virtualização. Ao se agrupar servidores lógicos distintos em uma única máquina física, consegue-se uma economia na aquisição de hardware e na manutenção de um parque computacional atualizado, além de utilizar-se menos espaço físico e consumir menos energia. Além disso, os servidores normais não raramente ficam ociosos entre 70% e 90% do tempo segundo Carmona (2008), e com a virtualização pode-se inverter esses indicadores.
- **Testes e desenvolvimento**: a virtualização permite desenvolver uma rede completa, inclusive com

[1] *Doutor em Bioinformática, Analista da Embrapa Informática Agropecuária, Campinas, SP, e-mail: lcintra@cnptia.embrapa.br*

hardware na realidade não disponíveis, para fins de testes. Também é muito utilizada na identificação de falhas durante o desenvolvimento do *kernel*[1] de sistemas operacionais.

- **Maior segurança e disponibilidade**: a segurança da rede pode ser melhorada quando utiliza-se servidores virtuais de um único propósito e blindados para cada serviço disponível. Por exemplo, um invasor não pode mais atingir o banco de dados de um servidor web comprometido, o que no modo de operação convencional seria difícil de ser evitado. A disponibilidade é incrementada, pois com a virtualização torna-se possível a existência de máquinas em *standby* que assumirão as funções na falha dos servidores principais. Além disso, se for necessário, pode-se transferir máquinas virtuais de um host para outro, e ali reiniciar os seus serviços.
- **Novas possibilidades para software**: com a virtualização, sistemas complexos poderiam ser entregues pelo fabricante, otimamente configurados em uma máquina virtual, minimizando assim o uso de tempo e pessoal nas organizações para a implantação de novos sistemas.
- **Sistemas legados**: a virtualização ajuda na execução de sistemas legados, permitindo que eles rodem em sistemas modernos.

Tipos de virtualização

Basicamente tem-se três técnicas diferentes para se chegar à virtualização de um sistema com um sistema operacional completo: a emulação (*emulation*), a virtualização total (*full virtualization*) e a paravirtualização (*paravirtualization*). Além dessas técnicas, tem-se ainda as virtualizações em nível de biblioteca e aplicação, apesar de elas não permitirem a execução de um sistema operacional completo, conforme indica Mattews et al. (2008). A seguir uma breve descrição de cada uma dessas técnicas.

Emulação

Na emulação, a máquina virtual simula completamente o hardware necessário para rodar os sistemas hóspedes, permitindo, inclusive, que arquiteturas distintas da arquitetura física em consideração sejam emuladas. A emulação é uma técnica útil quando se deseja testar sistemas operacionais para hardware que está em fase de desenvolvimento e ainda não se encontra fisicamente disponível. Perceba que os sistemas operacionais hóspedes não necessitam sofrer nenhuma alteração para executarem no ambiente virtual. Exemplos de sistemas que implementam tal técnica incluem o *PearPC*, *Bochs* e o *QEMU* na sua versão sem aceleração por hardware.

Virtualização total

Na virtualização total, os sistemas operacionais hóspedes também não necessitam sofrer nenhuma alteração para executarem, porém, os processadores das arquiteturas *x86* e *AMD64* devem dar suporte à virtualização. A AMD criou o projeto Pacifica ou *AMD Virtualization* (AMD-V) e a Intel; o projeto *Intel Virtualization Technology for x86* (Intel VT-x) foi para adicionarem suporte à virtualização em alguns de seus processadores.

Nessa modalidade de virtualização existe uma camada de software, chamada de *Virtual Machine Monitor* (VMM) ou *hypervisor*, entre o hardware e os sistemas operacionais, cuja função é gerenciar o compartilhamento dos recursos do sistema físico. Deve existir uma interface para se interagir com o *hypervisor*, e, no *Xen*, tal interface de interação com a VMM é disponibilizada em um sistema operacional com condições especiais chamado de Domínio 0 (*Dom0*), enquanto os sistemas hospedados são chamados de Domínio do usuário (*DomU*). Exemplos de sistemas de virtualização nessa modalidade incluem o *VMware ESX Server*, *Parallels Desktop*, *Win4Lin* e *Xen*.

Paravirtualização

Na paravirtualização, os sistemas hóspedes têm a consciência de que estão rodando em um ambiente virtualizado e, por esse motivo, devem ter seus *kernels* modificados para adquirirem esse comportamento. Por isso, o comum é ser encontrado apenas sistemas operacionais de código aberto rodando nessa modalidade. As maiores vantagens dessa abordagem são a performance, escalabilidade e a facilidade em manutenção. Exemplos de sistemas de virtualização que utilizam essa abordagem são o *User-mode Linux* (UML), o *Xen* e o *Vmware ESX*. O *hypervisor* exporta uma versão modificada do hardware sob seu controle, para cada sistema hospedado.

[1] O kernel *é a parte central de um sistema operacional e implementa as tarefas mais sensíveis e importantes. Também é chamado de núcleo, mas neste trabalho optou-se por manter o termo original, uma vez que o mesmo é bastante difundido em textos técnicos mesmo na língua portuguesa.*

Demais técnicas

Além dessas técnicas que permitem a virtualização de todo o ambiente, existem ainda outras técnicas que permitem uma virtualização parcial. O *Wine* é um exemplo de virtualização de bibliotecas, que tem a capacidade de rodar programas Windows sobre um ambiente Linux e, para tanto, ele propicia uma forma de executar as bibliotecas nativas do Windows.

Na virtualização de aplicações, uma máquina virtual é preparada para a execução de aplicações individuais, as quais são compiladas de maneira adequada. O principal exemplo dessa técnica é a *Java Virtual Machine* (JVM).

O *Hypervisor XEN*

Pode-se distinguir duas abordagens básicas para a implementação da virtualização. Na primeira abordagem, a camada de virtualização roda sobre um sistema operacional. Essa é a abordagem preferencial na virtualização de *desktops*. Ela aparece, por exemplo, no *VirtualBox* e *VMware Workstation*. Uma segunda abordagem é implementar a camada de virtualização diretamente sobre o hardware. Neste caso, a camada de virtualização é conhecida como *hypervisor* ou VMM. O *Xen* e o *VMware ESP* são exemplos de *hypervisor*. Neste trabalho, tem-se interesse na segunda abordagem.

Assim, para que se implemente um ambiente virtualizado com sucesso, é necessário uma forma de interação com o *hypervisor*. No caso do *Xen*, tem-se um sistema operacional que roda de maneira diferenciada e que irá possibilitar a comunicação necessária com a VMM, que é chamada de sistema de domínio 0 (*Dom0*), e quando se tratar de um sistema Linux, existirá um daemon chamado **Xend** que permite a comunicação com o *Xen* para a criação e manutenção das máquinas virtuais. Cada máquina virtual roda no domínio do usuário (*DomU*). Trata-se de uma alusão ao usuário que as aplicações rodam, justamente porque os sistemas operacionais rodando nesse nível não têm acesso direto ao hardware, e só o conseguem por meio do *hypervisor*. No *Dom0* são instaladas, também, uma série de ferramentas que servem de interface entre o usuário e o daemon *Xend*. Neste texto, essas ferramentas aparecerão e serão explicadas conforme forem necessárias, no entanto, o leitor não deve deixar de levar em consideração que existem muitas outras que não foram discutidas aqui.

Preparando um sistema *Dom0*

Como discutido anteriormente, o básico na construção de um ambiente de virtualização com o *Xen* é a instalação do *Dom0*, pois é por meio desse sistema que todas as máquinas virtuais serão controladas. Nesse ponto discutir-se-á a criação de um *Dom0* utilizando-se um sistema Linux, mais especificamente, utilizando-se o Ubuntu Linux.

Ainda, baseado no que apresentou-se até o presente momento, pode-se afirmar que para a obtenção de um *Dom0* rodando corretamente, precisa-se executar dois passos. Primeiramente instalar o *hypervisor Xen* propriamente dito e, em seguida, um *kernel* Linux preparado (modificado) para rodar sobre o *Xen*. A seguir discute-se cada um destes passos.

Antes, no entanto, para que se possa instalar o *Xen* em uma máquina, precisa-se que esta já tenha um sistema operacional em execução. Logicamente, esse será um sistema normal, não *xenificado*[2]. No presente caso, sugere-se que o sistema inicial seja um Ubuntu 9.10. Posteriormente, um *kernel* modificado com suporte ao Xen (*xenificado*) será compilado para, então, obter-se um *Dom0*. Atualmente, um *kernel* modificado para o *Xen* não está inserido diretamente nas principais distribuições Linux e a compilação de um *kernel* específico é a única forma de se trabalhar com a virtualização utilizando *Xen* nessas distribuições.

Instalando o *Xen* no Ubuntu

O *Xen* 3.3.0 está disponível nos pacotes do Ubuntu, e para instalá-lo basta executar o comando:

```
sudo apt-get install xen-utils-3.3
```

Ao final do processo de instalação, verifica-se que o arquivo *Xen-3.3.gz* está presente no diretório */boot*. Esse arquivo é o *hypervisor*. Na realidade, para se habilitar o *Dom0*, deve-se inicializar (*bootar*) esse arquivo e passar o *kernel* modificado como um parâmetro. Na sessão explica-se como alterar o grub para que isso aconteça.

Além de instalar o *hypervisor*, instale também um conjunto de ferramentas para se trabalhar com as máquinas virtuais. Para tanto, execute:

```
sudo apt-get install python
```

[2] *Compilando o* Kernel *Xenificado Corrente.*

Ao final do processo de instalação, ter-se-á disponível o comando **xm** que será utilizado intensamente na criação e manutenção das máquinas virtuais. Neste momento, se for executado o comando **sudo xm**, já se poderá perceber que ele está instalado. Mas, atenção, o comando ainda não funciona, pois faz comunicação com o *Xen* (o *hypervisor*) que ainda não está rodando.

Por fim, instale um pacote para a manipulação da *bridge* padrão do *Linux*, que é a forma padrão de comunicação das máquinas virtuais com o meio externo e, portanto, se você pretende utilizar essa característica do sistema, deverá instalar os pacotes:

```
sudo apt-get install libvirt0 libvirt-bin
```

Ao final do processo de instalação, execute o comando *ifconfig* para certificar-se que a interface *virbr0*, que é uma *bridge*, foi criada como esperado.

Compilando o *Kernel Xenificado* Corrente

Como mencionado anteriormente, o segundo passo no processo de obtenção de um *Dom0* é a disponibilização de um *kernel* que esteja adaptado para executar como tal. Esse *kernel* pode ser obtido já pré-compilado de algumas fontes na internet, mas ainda não está disponível no conjunto de pacotes do Ubuntu e distribuições similares. Por isso, neste trabalho, optou-se pela compilação do *kernel* (versão 2.6.32.x) que é mantido por *Jeremy* na árvore de *kernels* do Linux. Outra vantagem de se compilar o próprio *kernel* é com relação a um controle maior para futuros testes no sistema de virtualização. Então, primeiramente obtenha o kernel a partir dos repositórios oficiais:

```
cd /usr/src
sudo git clone -o xen -n git://git.kernel.org/
pub/scm/linux/kernel/git/jeremy/xen.git linux-2.6-
pvops.git
cd linux-2.6-pvops.git
git checkout -b xen/stable-2.6.32.x xen/xen/stable-
2.6.32.x
```

Neste momento, o *kernel* já está pronto para ser compilado e gerar o *Dom0*. No entanto, o seu processo de compilação exige que o Ubuntu tenha vários aplicativos e bibliotecas instaladas. Desta forma, para garantir que o sistema tenha os recursos necessários, instale os pacotes a seguir:

```
sudo apt-get install fakeroot build-essential
sudo apt-get install crash kexec-tools makedumpfile
sudo apt-get install git-core libncurses5 libncur-
ses5-dev
sudo apt-get install libcurl4-openssl-dev xserver-
xorg-dev mercurial gitk uuid-dev gawk gettext
sudo apt-get install texinfo bcc dpkg-dev debhel-
per iasl texinfo bridge-utils bison flex
```

Então, configure o *kernel* de acordo com suas necessidades. Em especial, habilite as opções *Xen* que interessam. Para tanto, execute os comandos abaixo e siga as instruções indicadas pelas figuras:

```
cd /usr/src/linux-2.6-pvops.git
sudo make menuconfig
```

Uma vez executado o comando **make menuconfig**, deverá aparecer um menu idêntico ao da Figura 1, que a partir de agora será referenciado como menu principal.

Para que se tenha um *kernel* habilitado como *Dom0*, deve-se executar dois passos simples. No primeiro passo, habilita-se o *kernel* como um *Dom0*, utilizando-se a opção a partir do menu principal **Processor type and features —> Paravirtualized guest support --> Enable Xen privileged domain support**. Para navegar pelo menu, utilize as teclas de movimentação e a tecla **enter**. Para selecionar/deselecionar um item como integrante do *kernel*, utilize a **barra-de-espaços**. Em seguida, volte ao menu principal utilizando a opção **Exit** do menu.

O segundo passo que deve ser executado é garantir que o módulo de *Bridge* esteja habilitado no *kernel*, pois o *Dom0* utilizará essa tecnologia para se comunicar com suas máquinas virtuais. Para tanto, a partir do menu principal, selecione **Networking support ----> Networking Options ----> 802.1d Ethernet Bridging**, e marque esse item como integrante do *kernel* (utilize a tecla barra-de-espaços até aparecer o símbolo *) ou como um módulo (símbolo M).

As configurações estão terminadas. Evidentemente que o *kernel* pode ser melhor otimizado, mas isso não será tratado aqui. Saia do menu de configurações salvando as opções selecionadas. A seguir, inicie a compilação do *kernel* com o comando:

```
sudo make
```

Uma vez que a compilação tenha ocorrido sem erros, instale os módulos do *kernel* e o próprio *kernel* com os comandos:

```
sudo make modules_install
sudo make install
```

Por fim, crie o arquivo **initrd** que será utilizado para a criação de um *RAM DISK* com os drivers integrados ao sistema durante o processo de inicialização.

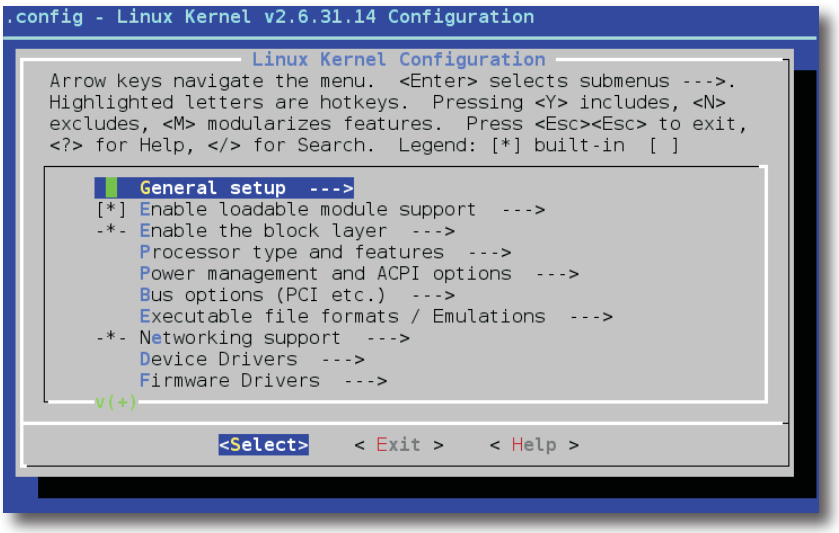

Figura 1. Janela com o menu principal.

```
sudo mkinitramfs -o /boot/initrd.img-2.6.32.24
2.6.32.24
```

Nesse ponto se tem um *Dom0* completamente funcional, bastando reinicializar o sistema para que o *Xen* comece a rodar. Evidentemente, é necessário que o *loader* atual tenha condições de inicializar o ambiente virtual corretamente durante o processo de inicialização (*boot*). Para aqueles que utilizam o *Grub*, a configuração necessária é obtida utilizando-se os seguintes comandos:

```
sudo apt-get install grub
sudo update-grub
sudo grub-install /dev/sda
```

Ao final desse processo, existirá no arquivo /boot/grub/menu.lst uma entrada semelhante a que aparece na Figura 2. Nesse caso, basta reiniciar o sistema no *Dom0*!

Instalando novos hóspedes

Em um ambiente de virtualização *Xen*, podem existir máquinas virtuais com virtualização total ou paravirtualizadas (PV). As máquinas virtuais com virtualização total utilizam uma camada de abstração chamada *Hardware Virtual Machine* (HVM) para interagir com as tecnologias de virtualização da AMD (*AMD-V*) e da Intel (*VT-x*) e por isso são conhecidas como máquinas HVM.

Instalando máquinas virtuais HVM

O processo de instalação de uma máquina virtual HVM é bastante simples, basta gerar um arquivo de configuração apropriado para essa máquina virtual e iniciar a sua execução, e a instalação decorre como em uma máquina real. Por padrão, os arquivos de configuração das máquinas virtuais ficam no diretório /etc/xen. A Figura 3 apresenta os dados de configuração de um arquivo que possibilita a instalação de um sistema *Windows 7*.

Nessa configuração, aparecem alguns itens que merecem ser explicados. Primeiramente, deve-se perceber que o programa que é carregado durante a inicialização da máquina virtual não é o *kernel* de um sistema operacional propriamente dito, mas, sim, um programa do *Xen* (o *hvmloader*), responsável por preparar o ambiente de virtualização total e posteriormente iniciar a execução do sistema operacional. Existe a linha de configuração **boot=cda** para indicar ao *hvmloader* que o sistema operacional deverá, primeiro, ser buscado no cd-rom (d), em seguida, no disco rígido (c) e, por fim, em um disquete (a). A linha *disk* permite indicar, ao sistema, quantos e quais serão os dispositivos de armazenamento que a máquina virtual possuirá. No presente exemplo, a máquina virtual contará com um HD que será criado em um arquivo no sistema de arquivos do *Dom0* (arquivo /home/xen/domains/windows7/disk1.img) e com um dispositivo de cd-rom que será criado a partir de uma imagem ISO contendo, não por acaso, o sistema operacional que deseja-se instalar.

```
title      Xen 3.3 / Ubuntu 10.04.1 LTS, kernel 2.6.31.14
uuid       4cf484b6-3a02-4610-ab88-c9f38069c4f7
kernel     /boot/xen-3.3.gz
module     /boot/vmlinuz-2.6.31.14 root=UUID=4cf484b6-3a02-4610-ab88-c9f38069c4f7 ro console=tty0
module     /boot/initrd.img-2.6.31.14
```

Figura 2. Entrada no arquivo de configuração menu.lst que permite a correta inicialização do ambiente virtualizado.

```
kernel = "/usr/lib/xen/boot/hvmloader"
builder="hvm"
device_model = "/usr/lib/xen/bin/qemu-dm"
memory = 768
name = "windows7"
vif = [ 'bridge=virbr0' ]
dhcp = "dhcp"
boot="dca"
disk = ['file:/home/xen/domains/windows7/disk1.img,hda,w',
        'tap:aio:/home/lcintra/ISOS/7100.0.090421-1700_x64fre_client_en-us_retail_ultimate-grc1culxfrer_en_dvd.iso,hdc:cdrom,r']
keymap="pt-br"
sdl=1
vnc=0
extra = 'xencons=xvc console=xvc0 video=xenfb'
#vfb = [ 'type=vnc,vncdisplay=1,vnclisten=127.0.0.1,vncpasswd=xenxenxen' ]

lcintra@d093:/etc/xen$
```

Figura 3. Conteúdo do arquivo de configuração /etc/xen/windows7.cfg, o qual possibilita a instalação de uma máquina virtual com o sistema operacional Windows 7.

Essa máquina virtual contará com 768 MB de memória e chamar-se-á windows7. O nome da máquina é um controle para que não existam mais de uma instância de uma máquina virtual (que leriam e gravariam informações em um mesmo dispositivo) rodando simultaneamente. Por fim, percebam a linha **sdl=1**. Essa linha de configuração cria um *framebuffer* (uma área na memória) de vídeo que permite a criação de uma janela no gerenciador de janelas do *Dom0*, a qual permitirá a interação com a máquina virtual; desde que o *kernel* do *Dom0* tenha sido compilado com suporte a *framebuffer*.

Nesse ponto, para que se possa inicializar a máquina virtual deve-se apenas providenciar o arquivo disk1.img. Nesse exemplo, utilizou-se a metodologia de armazenamento de máquina virtual mais simples possível. Desta forma, o disco da máquina virtual terá um tamanho fixo determinado pelo tamanho do arquivo disk1.img. Caso se desejasse um disco com a possibilidade de expansão ao longo do tempo, outras abordagens deveriam ser utilizadas, tais como o uso da ferramenta LVM (*Logical Volume Management*). Finalizando, deve-se providenciar um arquivo disk1.img com o tamanho desejado. Nesse exemplo, será um arquivo de 10 GB. Para tanto, utilize os comandos:

```
cd /home/xen/domains
mkdir windows7
sudo dd if=/dev/zero of=disk1.img bs=4098
count=2621440
```

Estando o arquivo pronto, inicialize a máquina virtual com o comando:

```
sudo xm create windows7.cfg
```

Irá aparecer a tela inicial de instalação do windows 7. Siga os passos necessários para completar a instalação. Ao final, a máquina virtual estará funcionando perfeitamente. Para um teste, execute o comando sudo xm list, o qual irá listar as máquinas virtuais em execução.

Máquinas virtuais PV

A instalação de máquinas virtuais PV não é tão simples quanto a instalação das máquinas HVM e existem muitas maneiras de ser feito, o que dificulta a discussão. Uma distinção importante que deve ser colocada inicialmente é o fato de sempre ser possível ter uma máquina virtual PV, cujo *kernel* esteja localizado no sistema de arquivos do sistema *Dom0*, assim como sempre é possível que o *kernel* esteja localizado diretamente no sistema de arquivos da própria máquina virtual. Nesse último caso, será necessário utilizar o *pygrub*, ferramenta disponibilizada pelo *Xen* para inicializar (bootar) a máquina virtual.

Vale ressaltar também que além das várias ferramentas existentes para a criação de máquinas virtuais PV, uma outra maneira eficiente de criar máquinas dessa natureza é instalando um sistema HVM, em seguida substitui-se o *kernel* desse novo sistema por um *kernel* que dê suporte à paravirtualização com o *Xen* e, então, modifica-se o arquivo de configuração no diretório /etc/xen para que o mesmo deixe de tratar a máquina virtual como uma HVM e passe a considerá-la como uma PV.

Uma das ferramentas disponíveis para a manipulação de máquinas virtuais e, em particular, para a instalação delas é o **xen-create-image** pertencente ao pacote **xen-tools**. Essa ferramenta tem a capacidade de instalar várias distribuições Linux diferentes: *debian: sid, sarge, etch, lenny; ubuntu: edgy, feisty, dapper; centos-4, centos-5, fedora-core-4, fedora-core-5, fedora-core-6 e fedora-core-7*. No entanto, deve ser mencionado que ela utiliza um *kernel* que esteja armazenado no sistema de arquivos do sistema *Dom0*. Dessa forma,

deve-se ter um *kernel* paravirtualizado para que a máquina virtual recém criada possa inicializar. A Figura 4, mostra o arquivo de configuração gerado para uma máquina virtual instalada com o auxílio da ferramenta **xen-create-image**.

Testes de desempenho nas máquinas virtuais

Evidentemente que quando pensa-se na virtualização, em algum momento, surge o questionamento do impacto que isso traz sobre o desempenho dos sistemas computacionais. Além disso, sabe-se que a virtualização total e a paravirtualização são implementações completamente distintas; que influenciam no desempenho final dos sistemas. Portanto, essa sessão apresenta algumas comparações iniciais do desempenho entre máquinas reais, o *Dom0*, máquinas totalmente virtualizadas e máquinas paravirtualizadas; e discute se as máquinas virtuais podem introduzir gargalos no desempenho de sistemas computacionais. Em especial, nesse ensaio, está se testando o *Dom0* porque ele é uma máquina virtual com características especiais e deseja-se analisar se isso o faria apresentar desempenho superior às demais máquinas virtuais. Os itens referentes aos quais se obteve testes de desempenho foram capacidade de processamento (cpu), capacidade de armazenamento de dados (leitura e escrita no HD) e desempenho de rede. A Tabela 1, resume os resultados obtidos com os testes.

Os testes de desempenho no armazenamento de dados foram realizados utilizando-se o aplicativo de *benchmark bonnie++*, exclusivo para essa finalidade. Investigou-se a capacidade dos sistemas em manipular grande quantidade de arquivos de pequeno

```
#
# Configuration file for the Xen instance teste, created
# by xen-tools 4.1 on Fri Oct 22 19:07:59 2010.
#
kernel      = '/boot/vmlinuz-2.6.32.24'
ramdisk     = '/boot/initrd.img-2.6.32.24'
memory      = '1024'
#
#  Disk device(s).
#
root        = '/dev/sda2 ro'
disk        = [
                  'file:/home/xen/domains/domains/teste/disk.img,sda2,w',
                  'file:/home/xen/domains/domains/teste/swap.img,hda1,w'
              ]
#  Hostname
name        = 'teste'

#  Networking
dhcp        = 'dhcp'
vif         = [ 'mac=00:16:3E:DB:F4:ED,bridge=virbr0' ]

sdl=1
#extra = 'xencons=xvc console=xvc0 video=xenfb'
#
#  Behaviour
#
on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

Figura 4. Imagem com a configuração gerado pela ferramenta **xen-create-image** durante a instalação de uma máquina virtual. As linhas finais, referentes aos dispositivos de interação com a VM foram acrescentadas manualmente.

Tabela 1. Tempos obtidos nos testes de *benchmark* com as máquinas virtuais.

Máquinas	*bonnie1*	*bonnie2*	*sysbench1*	*netperf1Dom0*
Dom0	1025,72s	306,17s	1118,5s	94,09 Mbits/seg
Virtualização total	1023,14s	508,88s	1117,89s	94,64 Mbits/seg
Paravirtualização	1023,52s	309,22s	1118,07s	94,51 Mbits/seg
Máquina real	1023,31s	283,43s	1116,95s	93,85 Mbits/seg

tamanho (teste *bonnie1*) e também poucos arquivos de grande tamanho (teste *bonnie2*). No primeiro teste (*bonnie1*), o programa de benchmark escreveu e leu 200 arquivos de 2 megabytes. Deve-se frisar que para cada teste de leitura e escrita, o programa *bonnie++* trabalha tanto com leitura/escrita caractere a caractere quanto em blocos de dados. Pode-se perceber, pelos resultados, que não há diferenças significativas entre o desempenho de máquinas virtuais e máquinas reais no caso de se trabalhar com arquivos de pequeno porte. Já no segundo teste (*bonnie2*), foram trabalhados 2 arquivos de 2 gigabytes cada. Esse é efetivamente o único teste no qual pode-se perceber uma perda de desempenho por parte das máquinas virtuais, e, nesse caso específico, deve-se observar que as máquinas paravirtuais têm melhor desempenho que as máquinas totalmente virtualizadas. Para mais detalhes sobre os comandos executados em cada teste, observe a Tabela 2.

Tabela 2. Comandos utilizados para os testes de *benchmark*.

Teste	Comando
bonnie1	bonnie++ -d /tmp -s 2M -n 0 -r 1G -x 200
bonnie2	bonnie++ -d /tmp -s 2G -n 0 -r 1G -x 2
sysbench1	sysbench --test=cpu --cpu-max-prime=200000 run
netperf1	netperf -H 10.129.1.1

O terceiro teste (*sysbench1*) diz respeito ao desempenho da cpu e consistiu em se usar o programa *sysbench* para encontrar os números primos menores que 200000. Não se observa nenhuma diferença significativa de desempenho entre as máquinas.

O mesmo ocorre para o quarto e último teste (*netperf1*), que utilizou o programa de *benchmark netperf* para avaliar o desempenho na transferência de dados pela rede. Novamente as máquinas virtuais não tiveram diferença de desempenho em relação às máquinas reais.

Dessa forma fica-se com a impressão de que as máquinas virtuais só terão perda de desempenho em tarefas que exijam a leitura e/ou escrita de grandes massas de dados.

Conclusão

A virtualização é, atualmente, uma ferramenta indispensável para uma administração eficiente de centros de dados, e as organizações, de modo geral, têm aplicado esforços para a sua implementação e disseminação. Atualmente, as principais vantagens apresentadas por essa metodologia consistem na redução do consumo energético (computação verde), otimização de espaço nos centros e aumento da disponibilidade dos sistemas. No entanto, muito trabalho tem sido dispensado para evolução dessa técnica e, certamente, grandes novidades surgirão em um futuro próximo.

Nesse cenário, dentre as tantas ferramentas existentes para a virtualização de servidores atualmente, o *Xen* se apresenta como uma forte concorrente. Dentre suas principais características facilitadoras para isso, pode-se citar o fato de ele ser de código-aberto, o que permite que organizações iniciem suas experiências com virtualização sem maiores investimentos; já ter implementado as principais funcionalidade para a virtualização e ter demonstrado sua capacidade de manter um ambiente de produção em inúmeras situações e organizações.

Dessa forma, este trabalho tem sua relevância, pois apresenta uma forma prática de se iniciar as atividades na virtualização de servidores com o *Xen* e demonstra que o desempenho das máquinas virtuais geradas não fica aquém das máquinas reais. Fica, ainda, como lição, que deve-se buscar o uso de máquinas paravirtualizadas, pois o desempenho destas é superior nas operações que exijam leitura e/ou escrita de grande quantidade de dados.

Referências

CARMONA, T. (Ed.). **Virtualização**. São Paulo: Linux New Media do Brasil, 2008. 319 p. (Coleção Linux technical review, v.1).

MATTEWS, J. N.; DOW, E. M.; DESHANE, T.; HU, W.; BONGIO, J.; WILBUR, P. F.; BRENDAN, J. **Running Xen**: a hands-on guide to the art of virtualization. : Upper Saddle River, NJ: Prentice Hall, 2008. 586 p.

Comunicado Técnico, 102

Embrapa Informática Agropecuária
Endereço: Caixa Postal 6041 - Barão Geraldo
13083-886 - Campinas, SP
Fone: (19) 3211-5700
Fax: (19) 3211-5754
http://www.cnptia.embrapa.br
e-mail: sac@cnptia.embrapa.com.br

Ministério da Agricultura, Pecuária e Abastecimento

1ª edição on-line - 2010

CGPE 9094