Comunicado Técnico 111

ISSN 1677-8464
Dezembro, 2011
Campinas, SP

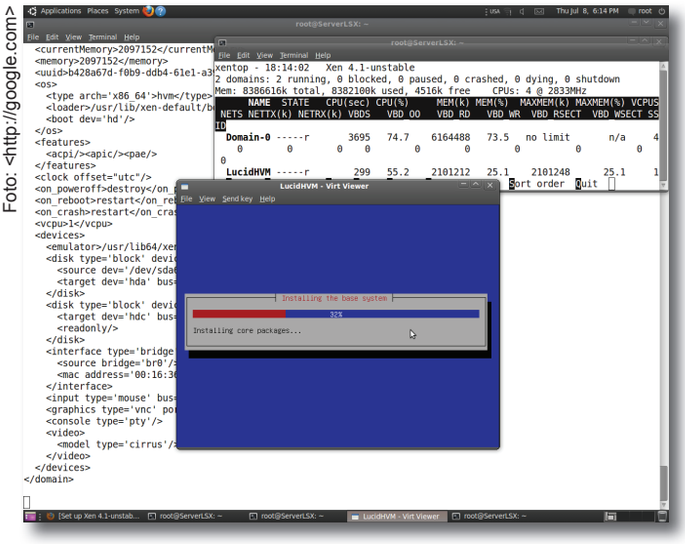

Foto: <http://google.com>

Uma proposta para a virtualização de servidores utilizando Xen

Leandro Carrijo Cintra[1]

O documento descreve uma proposta de virtualização para servidores que utilizam redundância de máquinas físicas e, por esse motivo, apresentam-se bastante resiliente a falhas. É descrito o processo de instalação de ambientes virtuais para servidores, utilizando o *hypervisor Xen* 4.1 e tendo-se um sistema Linux Ubuntu (*Natty release*) como *Dom0*. Com relação ao *DomU*, dá-se ênfase para máquinas virtuais paravirtualizadas (PV) que apresentam maior desempenho e flexibilidade em relação às máquinas totalmente virtualizadas (HVM).

Sobre a virtualização

A virtualização de servidores tem se difundido rapidamente e revolucionado a administração de *data centers* em todos os locais, não sendo portanto, um conceito que necessite aqui de muitas explicações. No entanto, é válido mencionar que a virtualização permite que vários sistemas operacionais distintos ou ambientes operacionais distintos que, por ventura, executem o mesmo tipo de SO, compartilhem um mesmo sistema físico. É uma tecnologia extremamente útil para racionalizar o uso de espaço físico e energia nas instalações de *data centers* e, com a introdução de sistemas físicos redundantes, tem a possibilidade de aumentar, consideravelmente, a resiliência de um sistema computacional.

Existem várias técnicas de virtualização aplicadas em computação, no entanto, quando se trata de virtualização de servidores, apenas duas são aplicáveis: a virtualização total (HVM) e a paravirtualização (PV). No primeiro caso, o SO virtualizado não tem conhecimento sobre sua condição, ou seja, ele executa exatamente como se estivesse sobre uma máquina real. Desta forma, uma série de operações desnecessárias, principalmente no que tange entrada/saída, serão executadas. Já na paravirtualização, os *drivers* do SO virtualizado são modificados para executarem, otimizadamente, sobre o ambiente virtual. Com isto, toda operação mais específica de entrada/saída não será simulada no DomU, mas os dados serão entregues ao Dom0 para que este realize diretamente a operação. Para maiores informações, consulte os documentos Mattews (2008), Brendel e Siqueira(2009) e Cintra (2010).

Nesses documentos, pode-se encontrar informações sobre as principais vantagens da virtualização, as quais são apenas citadas aqui: consolidação de servidores com economia de espaço e energia, testes de sistemas operacionais para arquiteturas em desenvolvimento,

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

Embrapa

maior segurança e disponibilidade, novas possibilidades relativas ao *deploy* de software e facilidades para se trabalhar com sistemas legados.

Há uma grande diferença entre a virtualização de *desktops* e de servidores. A primeira é conseguida com um software que executa como um processo sobre um determinado sistema operacional. Esse software fornece uma cópia virtual de cada um dos componentes de uma máquina física para os vários SOs que o mesmo gerencia. A segunda é conseguida com a implementação de uma camada de virtualização, um *hypervisor*, a qual é um software relativamente pequeno, que executa diretamente sobre o hardware, e assume completamente o seu controle. O *hypervisor* não implementa várias funções vitais de um SO e, dessa forma, deve sempre existir uma das máquinas virtuais (a chamada Dom0) cujo SO seja implementado de forma especial para complementar as funcionalidades da camada de virtualização, afim de que se obtenha um ambiente computacional efetivamente utilizável.

Existe uma grande gama de *hypervisor* disponível, e neste trabalho optou-se pela utilização do Xen 4.1; que implementa várias das principais características que podem ser obtidas em um ambiente virtual. Além disto, esse sistema é *open source*, o que facilita enormemente o seu acesso. Assim, a seguir, será discutida uma arquitetura robusta para um ambiente virtual, a instalação do Xen 4.1, a instalação de uma máquina virtual Dom0 utilizando o Ubuntu 11.04 e a instalação de máquinas paravirtualizadas, completando-se, assim, a implementação do ambiente pretendido.

Arquitetura para o ambiente virtual

Ao se configurar uma máquina virtual, existem três requisitos importantes no que se refere à localização das informações que ela precisa acessar. O primeiro deles é onde estará o arquivo de configuração dessa máquina, que possui todos os parâmetros necessários para o Xen criar a máquina virtual. O segundo é relativo à localização dos arquivos do sistema operacional. E o último requisito é onde se encontram os arquivos de dados. Compreender esses requisitos é vital quando se está planejando uma arquitetura fisicamente redundante, e deseja-se que as máquinas virtuais tenham liberdade para executarem em qualquer uma das máquinas físicas disponíveis.

Com relação ao arquivo de configuração, ele é um arquivo muito simples que deve estar presente no sistema de arquivos do Dom0, apenas sendo utilizado no momento de inicializar a máquina virtual e, justamente, essa é uma das tarefas do Dom0. Como cada máquina física tem o seu próprio Dom0, de alguma forma, os arquivos de configuração das máquinas virtuais têm de estar disponíveis em todas as máquinas físicas. Essa observação fica clara a seguir, quando é apresentada a arquitetura pretendida para o ambiente virtual.

Já os arquivos do sistema e os arquivos de dados fazem parte do sistema de arquivos da máquina virtual, e apesar de existir a hipótese destes serem armazenados em um arquivo no sistema de arquivos do Dom0, a melhor solução é que exista uma ou mais partições específicas para cada um desses conjuntos de arquivos. E novamente, as discussões a seguir a respeito da arquitetura deixarão claras quais as implicações desse fato.

Perceba que para se criar um ambiente onde as máquinas virtuais não dependam de uma determinada máquina física e possam rodar eventualmente em qualquer uma delas, basta que se garanta que todas as máquinas físicas tenham acesso aos arquivos de configuração e partições de sistema e dados das máquinas virtuais. Com certeza a forma mais segura de se implementar essa estratégia é utilizando um *storage* para armazenar as partições, tanto de sistema quanto de dados. Com relação aos arquivos de configuração, a solução ideal para eles depende da tecnologia de armazenamento adotada.

Na presente solução, utilizou-se como fonte para o armazenamento de dados um *storage area network* (SAN), mas nada impossibilitaria o uso de uma estrutura *network attached storage* (NAS). Muito sucintamente, a diferença entre as duas tecnologias é que no primeiro caso os sistemas operacionais (máquinas virtuais) conectados à infraestrutura recebem uma partição para o armazenamento, e são portanto, responsáveis pela manutenção do sistema de arquivos nessa partição. Para haver acesso concorrente a esses dados, é necessário que as máquinas virtuais implementem alguma metodologia de sincronização do acesso. No segundo caso, os sistemas operacionais (máquinas virtuais), recebem um sistema de arquivos; e desta forma, a responsabilidade por gerenciá-lo passa para o dispositivo NAS. Em particular, o acesso concorrente é muito facilitado, pois várias máquinas virtuais podem acessar o mesmo sistema de arquivos e as arbitragens necessárias ocorrerão por conta do dispositivo de armazenamento. Storages NAS são consideravelmente mais caros que storages SAN e as necessidades do projeto não justificavam a adoção do primeiro.

Voltando à questão dos arquivos de configuração, caso se utilize um sistema de armazenamento NAS, então, poderia haver uma partição única para o sistema Dom0 que seria compartilhada com todas as máquinas físicas e consequentemente, os arquivos de configuração de todas as máquinas virtuais estariam acessíveis a todas as máquinas físicas. No entanto, quando a tecnologia de armazenando é SAN, compartilhar uma mesma partição diretamente é impossível; e as soluções disponíveis para o compartilhamento não são adequadas. Poder-se-ia utilizar *network file system* (NFS), mas nesse caso introduz-se um *Single Point of Failure* (SPOF). Outra solução seria configurar a clusterização de dados para os Dom0s, mas, nesse caso, introduz-se considerável complexidade ao projeto para a solução de um problema simples. Uma forma elementar e eficiente de realizar o compartilhamento dos arquivos de configuração, é replicando-os em cada um dos Dom0. Desta forma, cada máquina física mantém uma ou mais partições exclusivas, nas quais estão instalado o ambiente Dom0. Como os arquivos de configuração mudam muito esporadicamente, podem ser sincronizados via um *script* utilizando-se *rsync*, por exemplo.

Estabelecidas essas observações, tem-se que uma arquitetura simples, porém eficiente, para o ambiente virtual, pode ser obtida utilizando-se um *storage* que possibilite a conexão simultânea de múltiplos servidores, os quais devem estar virtualizados com Xen. Tome como exemplo um *storage* que permita a conexão redundante de até quatro máquinas simultaneamente; e tem-se a representação da Figura 1 para a arquitetura proposta.

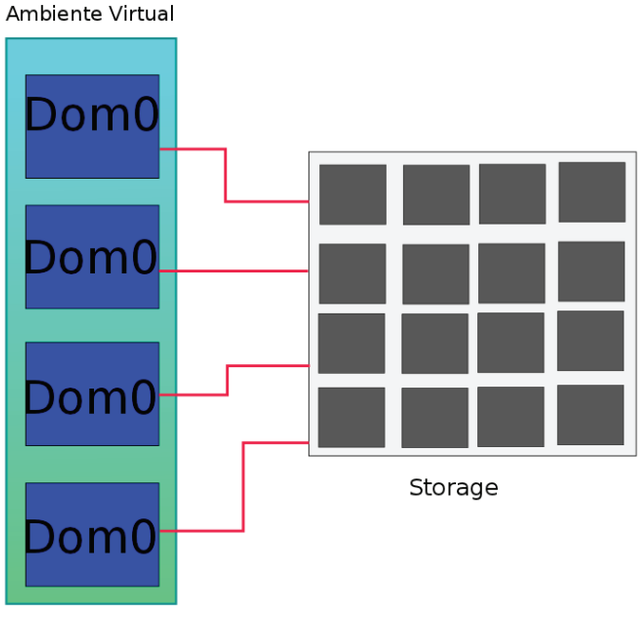

Figura 1. Ligações redundantes de quatro máquinas físicas ao storage, possibilitando que qualquer uma delas execute uma dada máquina virtual que utilize como fonte de informações partições do storage.

A configuração do ambiente não é complexa e depende basicamente da instalação do Xen e do Dom0 em cada uma das máquinas físicas, do adequado particionamento do *storage* para abrigar as partições de sistema e dados das máquinas virtuais e, finalmente, da configuração e instalação de cada uma delas.

Instalando o Xen no Ubuntu 11.04

Para que se possa usufruir das características mais avançadas da virtualização, tais como suporte à criação de *pools* de CPUs, compartilhamento de memória entre *guests* (máquinas virtuais), suporte à tolerância a falhas, migração de máquinas virtuais, dentre outros, é necessário que se trabalhe com a versão mais atual do Xen, ou seja, o Xen 4.1.2. A distribuição Ubuntu 11.04, ainda utiliza no repositório de pacotes o Xen 3.3. Desta forma, não é possível fazer a instalação do *hypervisor* por meio do sistema gerenciador de pacotes, o *Advanced Package Tool* (APT). Sendo assim, o caminho mais simples é obter o código fonte do Xen a partir do site oficial do projeto; e compilá-lo localmente. No entanto, o Xen possui algumas dependências, cujos pacotes disponíveis são apropriados, e possibilitam um bom funcionamento do ambiente. Assim, deve-se instalar o *python* e a biblioteca *libvirt* que é uma interface para diferentes sistemas de virtualização e oferece recursos corriqueiramente exigidos para a administração desses ambientes. Para tanto, basta executar os comandos:

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

Além dessas, existem outras dependências menos expressivas, mas que nem por isso podem deixar de estar presentes. Pode-se instalar todas elas com o comando:

```
  sudo apt-get install make gcc libc6-dev
zlib1g-dev libssl-dev uuid-dev libx11-dev
gettext libncurses5-dev bin86 bcc libc6-
-dev libc6-dev-i386 iasl python-dev te-
xinfo texlive transfig ocaml-findlib
```

O processo de instalação do Xen também é muito simples. Primeiramente obtenha o seu código fonte, navegando pelo site oficial do projeto (http://xen.org) ou simplesmente utilizando o comando:

```
wget http://bits.xensource.com/oss-xen/
release/4.1.2/xen-4.1.2.tar.gz
```

Uma vez tendo o arquivo, descompacte-o para iniciar os trabalhos de compilação. Uma boa prática é manter o código do Xen no diretório `/usr/src` que é a localização padrão para o código fonte do kernel Linux. Como o Xen exerce muitas atividades de um sistema operacional, nada mais natural que o seu código fonte esteja nesse diretório. Para descompactar o arquivo execute:

```
cd /usr/src
sudo tar -xvzf /tmp/xen-4.1.2.tar.gz
```

Finalmente, compile o sistema com os comandos:

```
sudo make xen
sudo make tools
sudo make install-xen
sudo make install-tools
```

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 este está instalado. Mas atenção, o comando ainda não pode realizar suas tarefas corretamente, pois o mesmo precisa comunicar-se com o *Xen* (o *hypervisor*) que ainda não está executando. Além disso, no diretório /boot estará presente o arquivo Xen-4.1.2.gz que é exatamente o binário do Xen, que será carregado durante o processo de inicialização da máquina física e assumirá o seu controle.

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. Outrora fora necessário a compilação do kernel Linux modificado para dar suporte ao domínio 0. Atualmente, quando se utiliza a distribuição Ubuntu, ou qualquer kernel Linux a partir da versão 3, isso não é mais necessário uma vez que o kernel presente já está "xenificado", ou seja, já traz consigo suporte ao Xen; portanto, pode ser utilizado como Dom0. Enfatizando, isso significa que o Xen poderá inicializar o kernel Linux presente no Ubuntu sem nenhum problema, pois o mesmo já está habilitado a executar sobre a camada virtual. Para aqueles que desejam compilar seu próprio kernel com suporte ao Xen, busquem mais informações em Cintra (2010).

Desta forma, uma vez que o Xen já está instalado e existe um kernel compatível com ele, basta que a máquina seja reinicializada para que o ambiente virtual estabeleça-se. Antes, no entanto, o sistema gerenciador de *boot* deverá ser informado das atualizações e que, a partir de então, é o Xen que deverá ser carregado durante o processo de inicialização. Quando se utiliza o *Grub* como gerenciador de *boot*, isso pode ser conseguido com o comando:

```
sudo update-grub
```

Esse comando irá buscar pelos *kernels* presentes no diretório /boot e gerar uma entrada no arquivo de configuração do *Grub* para cada um deles. Além disso, ele irá identificar a presença do *Xen* e dos kernels xenificados e irá gerar uma entrada apropriada para cada um. Uma entrada para a camada virtual no arquivo de configuração do Grub deverá indicar que o arquivo a ser carregado é o Xen-4.1.2.gz e o kernel Linux xenificado é passado como um dos parâmetros da operação de carga.

Finalmente, reinicie o sistema e no menu de seleção do Grub e escolha uma das opções com o Xen. Ao final do processo de inicialização, o comando abaixo poderá ser utilizado para se testar se o ambiente virtual está operando adequadamente.

```
sudo xm list
```

O processo de instalação do Xen deverá ser repetido em cada uma das máquinas físicas que farão parte da infraestrutura de virtualização.

Instalando um DomU PV com o debootstrap

Nesse documento, omitiu-se a configuração do *storage* no Dom0. Isto ocorre porque nem sempre o armazenamento de dados é feito em um *storage* e, também, porque o processo de configuração é dependente do seu modelo e não faz muito sentido apresentar aqui os passos para um caso específico. Melhor é considerar uma abstração para essa etapa, que pressupõe que exista uma partição que a máquina Dom0 possa utilizar para criar a partição de dados da máquina paravirtualizada prestes a ser instalada. Caso a infraestrutura física seja redundante e, portanto, tenha várias máquinas físicas, é necessário que essa partição seja visível em cada uma delas. No presente caso, o *storage* exporta a partição para todas as máquinas físicas por

meio do protocolo *iSCSI*. A mesma solução pode ser obtida com uma máquina dedicada ao armazenamento de dados, mas que não seja um *storage*. Não faz diferença qual dos Dom0 realmente irá trabalhar para a instalação do sistema paravirtualizado na partição. Depois de completa a instalação, a MV poderá executar em qualquer máquina física, desde que só exista uma instância dela executando a cada momento; ou que as máquinas virtuais tenham sido configuradas em *cluster*.

Existem várias formas e ferramentas para a instalação de máquinas paravirtualizadas. O debootstrap é uma ferramenta disponível para a criação de uma instalação com os pacotes essenciais de um sistema Debian/Ubuntu, que é o desejado nesse projeto. Dessa forma, a seguir será descrito o processo para a instalação utilizando essa ferramenta.

Inicialmente, é necessário montar a partição onde se deseja instalar o sistema. Como o nome dessa partição mudará de ambiente para ambiente, optou-se aqui por chamá-la de yourPartition:

```
  mkdir /tmp/PV
  sudo mount yourPartition /tmp/PV
  sudo debootstrap --arch=amd64 –
include=linux-image-generic,libc6,grub
--components= main, universe, multiverse
natty /tmp/PV http://http://us.archive.
ubuntu.com/ubuntu/
```

Uma vez que o *debootstrap* tenha instalado os arquivos de sistema básicos, os passos seguintes exigem que o diretório de instalação torne-se o diretório raiz. Isso pode ser conseguido com o chroot. Dessa forma:

```
  sudo chroot /tmp/PV
  export LANG=C
```

Crie o arquivo /etc/fstab com o conteúdo abaixo:

```
  #cat /etc/fstab
  /dev/xvda1 / ext4 defaults   0 1
  proc /proc proc defaults     0 0
  update-grub
```

Edite o arquivo /boot/grub/menu.lst e remova os parâmetros `quit splash` do kernel e adicione o parâmetro `console=xvc0`.

Edite o arquivo /etc/init/tty1.conf e substitua tty1 por xvc0. Também adicione xvc0 no arquivo /etc/securetty para permitir o login do root na console xvc0.

Também remova qualquer referência para o clock do hardware:

```
  update-rc.d -f hwclockfirst remove
  update-rc.d -f hwclock remove
  rm /etc/udev/rules.d/85-hwclock.rules
```

É necessário também configurar a interface de rede no arquivo /etc/network/interfaces, informações sobre o host no arquivo /etc/hosts e o nome da máquina no arquivo /etc/hostname. Finalmente, pode-se deixar o sistema de arquivos do DomU:

```
  exit
  umount /tmp/PV
```

Para finalizar, basta que o arquivo de configuração da máquina virtual seja criado no sistema de arquivos do Dom0. Uma vez que instalou-se um kernel na partição de sistema do DomU, pode-se fazer uso do aplicativo pygrub que possibilita que esse kernel seja utilizado durante o processo de inicialização da máquina virtual. A forma mais simples de se criar esse arquivo é partir de um dos exemplos disponíveis no diretório /etc/xen e realizar as devidas adaptações. A Figura 2a mostra as principais linhas de configuração, após as adaptações e a exclusão da maioria dos comentários. Outra opção é utilizar um kernel que esteja armazenado no sistema de arquivos da máquina Dom0. Neste caso, o arquivo de configuração ficaria como ilustrado pela Figura 2b.

Migrando máquinas virtuais

A grande vantagem do ambiente com unidades físicas redundantes tendo acesso à mesma fonte de armazenamento é que as máquinas virtuais tornam-se independentes do hardware e podem ser migradas de acordo com as necessidades. Basicamente, essa migração pode dar-se de quatro formas:

a) pode-se, simplesmente, parar manualmente uma máquina virtual em uma máquina física e reiniciá-la em outra. Nesse caso, não há a necessidade de nenhuma configuração especial além das configurações já discutidas. Também não há a necessidade de um aplicativo específico;
b) pode-se migrar on-line uma máquina virtual em execução *live migration* entre máquinas físicas. Nesse caso, é necessário apenas que o *hypervisor* esteja configurado para aceitar tal operação;

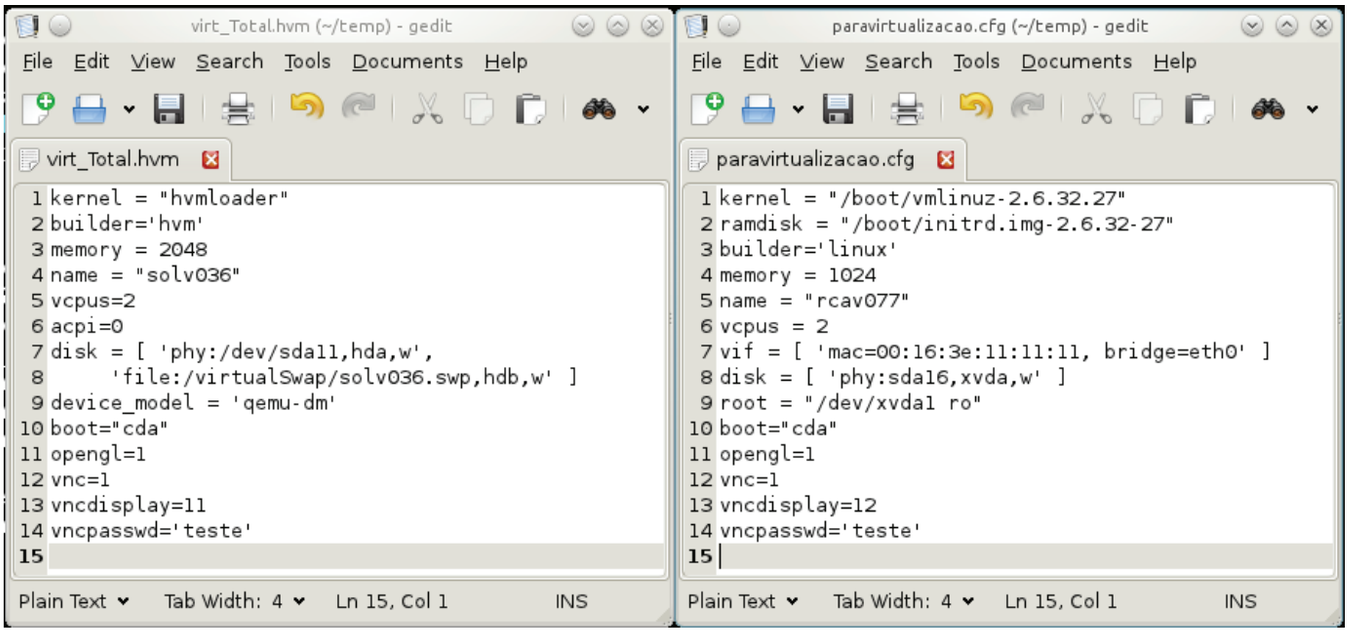

Figura 2. Arquivos básicos de configuração de um guest totalmente virtualizado (esquerda) e paravirtual (direita).

c) pode-se configurar um ambiente que detecte a falha de uma máquina virtual e automaticamente, a reinicie em outra máquina física. Nesse caso, será necessário um sistema que execute tal tarefa;
d) e finalmente, pode-se configurar um ambiente em que coexistam, simultaneamente, duas máquinas virtuais idênticas, uma executando e outra em *standby*, sendo que esta última só entrará em execução em caso de falha da primeira. Para tanto, o Xen utiliza o *Remus* (CULLY et al. 2008), um sistema que implementa alta disponibilidade de forma transparente para as aplicações.

A configuração do Xen para a migração de máquinas virtuais, mencionada no segundo caso acima, é simples e dependente apenas de se garantir que a linha

```
#(xend-relocation-hosts-allow '^localhost$ ^localhost\\.localdomain$')
```

esteja comentada no arquivo /etc/xen/xend-config.sxp, enquanto a linha

```
(xend-relocation-port 8002)
```

deve estar, obrigatoriamente, descomentada. Uma vez que o xend, *daemon* reponsável por estabelecer a comunicação entre o Dom0 e o *hypervisor* para as operações de gerenciamento das máquinas virtuais, tenha sido reiniciado com o comando:

```
sudo /etc/init.d/xend restart
```

as máquinas virtuais poderão ser manualmente migradas on-line entre máquinas físicas com o comando

```
sudo xm migration <nomeVM> <nomeHost-Destino> --live
```

O estabelecimento de um procedimento que possibilite a reinicialização automática das máquinas virtuais quando uma máquina física falhar exige, como mencionado na terceira forma de migração acima, a instalação e configuração de um sistema independente do Xen. Uma possibilidade é se usar o sofware *heartbeat*, que implementa a alta disponibilização de recursos de forma já bastante desenvolvida e estável. Uma vez que se instale e configure o sistema corretamente, este ficará monitorando as máquinas virtuais e caso alguma deixe de executar, ela será restabelecida em outra máquina física.

Conclusão

Não há dúvidas que a virtualização é uma tecnologia com um grande potencial de uso atualmente, em virtude das várias boas características que ela apresenta, mas principalmente pela possibilidade da redução de custos com infraestrutura de TI e o uso mais racional dos recursos energéticos, o que na atualidade, é um fator muito importante.

O Xen se apresenta como uma excelente possibilidade para o uso da tecnologia por parte das organizações, uma vez que ele implementa as principais ferramentas disponíveis no campo da virtualização e tem a licença GPL (*General Public License*), o que possibilita um início dos trabalhos com baixo investimento.

Neste trabalho, abordou-se uma situação onde a virtualização passa a ser considerada mais seriamente dentro da organização; e presentou-se uma arquitetura que pode ser implantada sem muitas dificuldades, apresentando ,como principal característica, a facilidade para recuperar os sistemas em caso de falhas físicas.

Referências

BRENDEL, J.; SIQUEIRA, L. A. **Virtualização**: Linux New Media, 2009.

CINTRA, L. C. Virtualização com o Xen: instalando e configurando o ambiente. Campinas: Embrapa Informática Agropecuária, 2010. 8 p. (Embrapa Informática Agropecuária. Comunicado Técnico, 102). Disponível em: <http://ainfo.cnptia.embrapa.br/digital/bitstream/item/31526/1/ct102-10.pdf>. Acesso em: 1 dez. 2011.

CULLY, B.; LEFEBVRE, G.; MEYER, D.; FEELEY, M.; HUTCHINSON, N.; WARFIELD, A.; Remus: high availability via asynchronous virtual machine replication. In: USENIX SYMPOSIUM ON NETWORKED SYSTEMS DESIGN AND IMPLEMENTATION, 5., 2008, San Francisco. **Proceedings**... [S.l.]: Usenix, 2008. p. 161-174. NSDI '08.

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: Prentice Hall, 2008. 586 p.

Comunicado Técnico, 111

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

1ª edição on-line - 2011

CGPE 9790