Capítulo 1

Xen — Básico sobre Plano de Fundo e Virtualização

O Xen é um monitor de máquina virtual (*hipervisor*) que permite que se utilize um computador físico para executar muitos computadores virtuais – por exemplo, executar um servidor de Internet em produção e um de teste na mesma máquina física ou executar Linux e Windows simultaneamente. Embora não seja o único sistema de virtualização disponível, o Xen tem uma combinação de características que o tornam especialmente adequado para muitas aplicações importantes. O Xen executa em plataformas de hardware comuns e é um software de código aberto. Ele é rápido, pode-se facilmente mudar sua escala de operação, e possui características típicas de servidor, como, por exemplo, migração ativa. Este capítulo discute tipos e usos comuns para a virtualização, descreve a história desta e as origens do Xen, traz uma visão geral e sucinta sobre a estrutura do Xen e o compara a outros sistemas de virtualização.

Usos Comuns e Benefícios da Virtualização

Monitores de Máquinas Virtuais são uma forma conveniente de usar o mesmo computador físico para executar muitas tarefas diferentes. Há anos que sistemas operacionais têm feito isso simplesmente permitindo que usuários executem múltiplos aplicativos ao mesmo tempo, como, por exemplo, navegadores de Internet, servidores de base de dados e jogos. No entanto, sem a virtualização, o ato de escolher um sistema operacional e configuração de sistema para executar em seu computador físico tem o infeliz efeito colateral de limitar muitas outras opções. Por exemplo, se o Linux estiver sendo executado para desenvolvimento e teste de aplicativos, pode ser impossível executar programas escritos exclusivamente para Windows. Como outro exemplo, se a última versão totalmente atualizada do Windows estiver sendo executada, pode ser difícil reproduzir problemas de clientes executando versões mais antigas. Mais: se um servidor de Internet e outro de base de dados exigem versões diferentes de uma biblioteca de sistema, pode ser impossível executar os dois num mesmo sistema. Sem virtualização em cada um desses exemplos, seria necessário manter muitas máquinas físicas, cada uma com sua configuração específica de software, mesmo se os recursos de computação em uma máquina forem suficientes para executar todos os aplicativos simultaneamente.

Os monitores de máquinas virtuais (*hipervisores*) estão se tornando cada vez mais importantes na computação moderna, porque permitem que diferentes sistemas operacionais e configurações coexistam na mesma máquina física. O hipervisor controla o hardware subjacente, permitindo que ele seja usado por muitos sistemas hóspedes (guests systems, literalmente "sistemas convidados") ao mesmo tempo e dando a cada sistema hóspede a ilusão de que ele está executando num hardware privativo.

O hipervisor abstrai os recursos físicos do computador hospedeiro (host), disfarçando-os como contrapartidas virtuais discretas que podem ser alocadas para uso por hóspedes individuais. Hóspedes virtuais tratam seu hardware virtual como se ele fosse real, e o hipervisor garante que essa ilusão seja perfeita. Além disso, os hipervisores precisam garantir algum nível de isolamento entre os hóspedes. De certa forma, eles agem tanto como prestidigitadores quanto como guardas de trânsito. A Figura 1.1 ilustra o relacionamento entre o hardware físico, o hipervisor e as máquinas virtuais hóspedes.

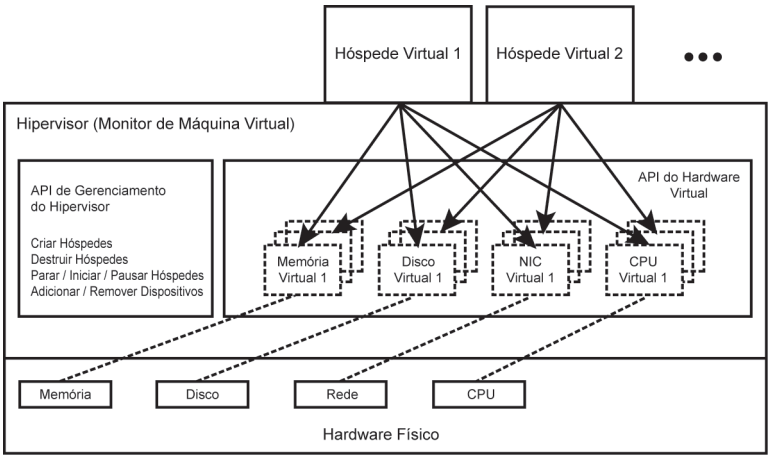

FIGURA 1.1 – O hipervisor fica entre os domínios hóspedes e o hardware físico.

Monitores de máquinas virtuais também permitem uma interface uniforme com o hardware. Isso protege os sistemas hóspedes de alguns detalhes de mais baixo nível dos recursos físicos de computação e permite portabilidade, que é outro benefício-chave da virtualização. De fato, muitos hipervisores modernos permitem que seus sistemas hospedados sejam deslocados de uma máquina física para outra sem nenhuma interrupção. As configurações desses sistemas podem ser facilmente desenvolvidas numa máquina e então distribuídas para diversas outras. Isso diminui o trabalho de gerenciar e distribuir software em coleções de máquinas com características de hardware diferentes. Sistemas hóspedes podem mesmo migrar de um computador físico para outro enquanto executam. O Xen chama isso de *migração ativa* (live migration, literalmente "migração viva" ou "ao vivo"). Alguns dos benefícios da virtualização são:

- Depurar sistemas operacionais consome muito tempo e exige programação de qualidade excepcional. A virtualização pode diminuir tal fardo ao permitir que um desenvolvedor teste novos sistemas operacionais como hóspedes em ambientes mais estáveis. Há muitos anos essa técnica tem sido usada e tem se mostrado eficiente. De forma semelhante, pesquisadores de segurança podem criar sistemas operacionais hóspedes que ficam isolados de outros e mesmo da máquina que o hospeda. Esse tipo de sistema permite que os pesquisadores estudem os efeitos de worms, trojans e de vírus sem afetar o sistema hospe-

deiro. Um termo coloquial para explicar isso é dizer que o sistema está numa "sandbox" (literalmente "caixa de areia"). Um hóspede numa sandbox pode ser também usado para testar atualizações ou softwares experimentais de ponta antes de aplicá-los a sistemas em produção.

- Outro benefício da virtualização é a capacidade de se recuperar rapidamente de problemas de software causados por ataques maliciosos deliberados ou por falhas acidentais de funcionamento. Ao manter uma cópia estável de um hóspede, se recuperar de um ataque pode ser tão simples quanto retornar para esse ponto confiável salvo.
- Virtualização pode resultar em alta disponibilidade realocando hóspedes quando uma máquina servidora se torna indisponível. Ambientes servidores podem ser compostos de muitas máquinas físicas, cada uma executando uma série de sistemas hóspedes. Estes podem ser movidos entre máquinas físicas de maneira imperceptível para os usuários a fim de balancear carga dinamicamente, usando dessa forma recursos agregados de maneira mais eficiente. Há muitos anos, diversas empresas têm se aproveitado de benefícios como esse em plataformas exóticas de hardware. O Xen agora disponibiliza essas vantagens para uma maior gama de usuários.
- Outro benefício da virtualização se torna especialmente claro num ambiente servidor. Um exemplo é a capacidade de consolidar muitos serviços numa única máquina física enquanto ainda se permite que cada um deles seja administrado independentemente. Num ambiente hospedando múltiplos sistemas, um provedor de serviços pode executar sistemas hóspedes, pertencendo a muitos indivíduos ou empresas na mesma máquina física. Cada entidade pode ter seu próprio acesso à raiz ou administrativo, fazer suas próprias escolhas a respeito de quais softwares executar e administrar seu próprio espaço de forma autônoma, sem a necessidade de consultar ou coordenar com os proprietários dos demais sistemas.
- A maior parte das vantagens da virtualização, especialmente em plataformas comuns como a x86, derivam da abundância do poder de computação disponível numa única máquina. Conforme o poder de um sistema médio vai crescendo, também vai a medida de capacidade de computação subutilizada – especialmente em sistemas com múltiplos processadores ou com processadores de múltiplos núcleos. A virtualização serve como forma de se aproveitar dessa capacidade de computação latente se apoiando nas máquinas cada vez mais poderosas.

- Hipervisores podem ser especialmente úteis para desenvolvedores, porque com eles a necessidade de reiniciar máquinas físicas para alternar entre vários sistemas operacionais desaparece. Configurações para inicializar diversos sistemas não são mais suficientes para esses desenvolvedores. A necessidade dessa funcionalidade está se tornando mais comum conforme mais aplicativos são desenvolvidos para serem distribuídos em múltiplas plataformas.

De uma perspectiva empresarial, virtualização pode permitir uma redução de custos totais de propriedade (TCO, total cost of ownership). O hardware é mais bem aproveitado quando múltiplos sistemas operacionais coexistem numa única máquina física. Imagine executar apenas duas máquinas virtuais em cada servidor que uma empresa possui. Isso poderia representar que apenas 50% do hardware seria necessário para uma infra-estrutura de computação equivalente. Veja que não estamos sugerindo aqui que todos os computadores deveriam estar executando sistemas operacionais virtuais hóspedes simultâneos, mas freqüentemente muitos computadores permanecem inativos, e estes são os candidatos mais indicados para consolidação via virtualização. Custos de treinamento para empregados podem ser reduzidos quando se utiliza virtualização porque permite que muitas configurações de treinamento diferentes (de sistemas operacionais e aplicativos) coexistam numa plataforma única, reduzindo dessa forma a quantidade de computadores necessários para treinamento, e a reconfiguração também será mínima entre diferentes seções.

- Em muitos cenários empresariais, usuários podem ter como vantagem a capacidade de executar sistemas operacionais e aplicativos antigos em plataformas de hardware modernas. Tipicamente a migração de tais aplicativos para arquiteturas atuais tem custo alto. E, mesmo se ela for bem-sucedida, esses aplicativos novos terão de ser depurados por muitos anos para se tornarem robustos como os programas originais. Com uma máquina virtual, os usuários podem executar produtos antigos livremente num ambiente virtual protegido sem temer que algum aplicativo antigo com problemas faça com que todo o sistema pare.
- O benefício final da virtualização que vale a pena mencionar é o consumo de potência reduzido e a menor necessidade por uma infra-estrutura de resfriamento. Servidores executando virtualização com alta taxa de utilização fazem melhor uso da potência elétrica do que muitos sistemas com baixa taxa. Quanto menos espaço é ocupado pela infra-estrutura de computação, sobrará mais espaço para resfriar adequadamente os centros de dados muito densos e quentes da atualidade. Em alguns casos pode-se poupar somas substanciais com ar condicionado.

Tipos de Virtualização

Muitos detalhes técnicos a respeito de virtualização são similares, e ainda assim existem muitas abordagens para resolver problemas associados com diferentes implementações. Quatro principais arquiteturas para virtualização na computação moderna permitem a ilusão de sistemas isolados: emuladores, virtualização completa, paravirtualização e virtualização em nível de sistema operacional. Para fazer uma discussão mais completa, também discutiremos brevemente dois outros tipos, virtualização de bibliotecas e de aplicativos, mesmo que estes não sejam capazes de executar sistemas isolados completos com seus respectivos sistemas operacionais.

Sistemas operacionais modernos em computadores pessoais geralmente permitem uma forma frágil de isolamento através do uso de processos individuais, com generosas capacidades de compartilhamento de dados entre eles. Muitos PCs são projetados para utilização por um único usuário, de forma que compartilhamento tem geralmente precedência sobre o isolamento. Um PC moderno pode ter qualquer número de programas executando como processos separados. Cada um tem seu próprio identificador exclusivo de processo obtido de uma reserva global, mas compartilha um sistema de arquivos únicos subjacentes. Em contraste, hipervisores são projetados para conseguir um isolamento muito mais forte entre máquinas virtuais. A maioria dos hipervisores não permite mais compartilhamento entre instâncias de sistemas hóspedes do que computadores separados numa mesma rede.

Cada técnica de virtualização efetua uma troca entre nível de isolamento e aumento de compartilhamento de recursos entre os hóspedes. Tipicamente, isolamento mais forte tem como custo alguma diminuição de performance. Isso é devido ao processamento extra necessário para a implementação de mecanismos de isolamento fortes. De forma inversa, isolamento mais fraco pode diminuir os requisitos de implementação e assim melhorar a performance.

Emuladores

Nos emuladores, a máquina virtual simula todo o conjunto de hardware necessário para executar hóspedes sem que nenhuma modificação seja necessária, para diferentes arquiteturas de hardware. Isso é ilustrado na Figura 1.2. Tipicamente, emuladores são usados para criar novos sistemas operacionais ou microcódigo para novos projetos de hardware antes que estes estejam disponíveis fisicamente. Exemplos incluem PearPC, Bochs e as formas não aceleradas do QEMU.

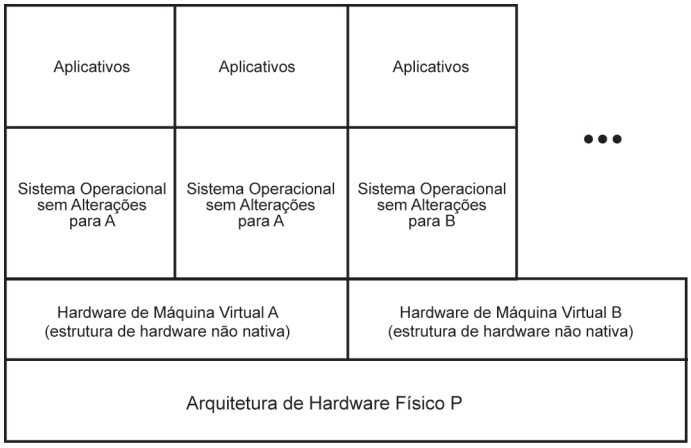

FIGURA 1.2 Máquinas virtuais emuladas simulam uma arquitetura de computação virtual que não é a mesma que a arquitetura física da máquina hospedeira. Sistemas operacionais destinados a funcionar no hardware emulado executam sem modificações.

Virtualização Completa

A *virtualização completa* (também chamada de *virtualização nativa*) é semelhante aos emuladores. Como neles, sistemas operacionais sem modificação executam dentro de uma máquina virtual. A diferença em relação aos emuladores é que sistemas operacionais e aplicativos são projetados para executar na mesma arquitetura de hardware presente na máquina física subjacente. Isso permite que um sistema em virtualização completa execute muitas instruções diretamente no hardware bruto. O hipervisor, neste caso, vigia o acesso ao hardware subjacente e dá a cada sistema operacional hóspede a ilusão de ter sua própria cópia desse hardware. Não é preciso usar software para simular uma arquitetura básica diferente. A Figura 1.3 ilustra a virtualização completa.

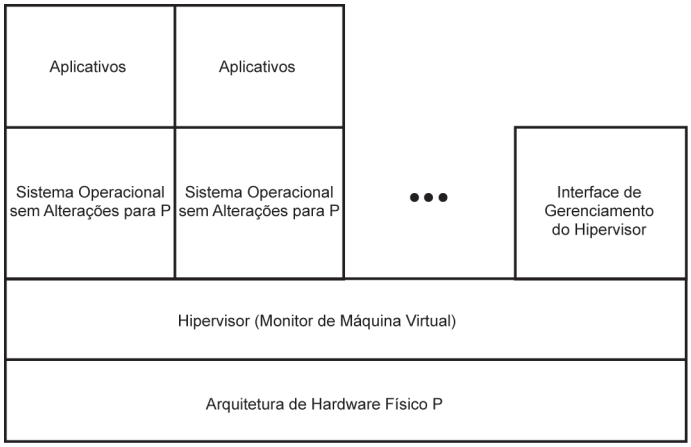

FIGURA 1.3 Os hipervisores em virtualização completa apresentam o hardware físico efetivo "P" para cada hóspede de forma que sistemas operacionais destinados a funcionar na estrutura subjacente possam executar sem modificação e sem perceber que estão sendo virtualizados.

Para os x86, sistemas são freqüentemente classificados como de virtualização completa se podem executar os arquivos binários dos sistemas operacionais sem modificações. No entanto, alguns desses ainda fazem algumas alterações simplificadoras nos x86 para facilitar a virtualização e ainda assim conseguir alta performance. A arquitetura x86 é reconhecidamente difícil de virtualizar. Por esse motivo, dados específicos sobre virtualização (o VT da Intel e o AMD-V da AMD, discutidos em seções com seus nomes no Capítulo 4, "Requisitos de Hardware e Instalação do Domain0 do Xen") foram acrescentados para melhorar a performance e tornar mais simples a execução de sistemas operacionais dentro da máquina virtual do Xen. O suporte para esses módulos é feito através de métodos inteligentes, como, por exemplo, a tradução de instruções que não se desejam nas versões simplificadas da arquitetura x86, feita nos arquivos binários conforme são executados.

Produtos importantes nessa área incluem os da VMWare, com seus Workstation (estação de trabalho) e Server (servidor, antes conhecido como GSX Server), os da Parallels Desktops (literalmente "áreas de trabalho paralelas"), o Win4Lin Pro e o z/VM. O Xen suporta virtualização completa em arquiteturas básicas com o suporte de hardware para a virtualização previamente mencionado.

Paravirtualização

Uma terceira técnica comum é conhecida como *paravirtualização*. Em alguns lugares ela é também chamada de iluminação (enlightenment). Nela, o hipervisor exporta uma versão modificada do hardware físico subjacente. A máquina virtual exportada é da mesma arquitetura, o que não é necessariamente o caso dos emuladores. Em vez disso, modificações específicas são introduzidas para facilitar e acelerar o suporte a múltiplos sistemas operacionais hóspedes. Por exemplo, um sistema hóspede pode ser modificado para usar uma interface binária de aplicativo (ABI, Application Binary Interface) de hiperchamada, em vez de usar certas características da arquitetura que seriam normalmente empregadas. Isso quer dizer que apenas pequenas mudanças são tipicamente necessárias nos sistemas operacionais hóspedes, mas mesmo assim elas tornam difícil dar suporte para sistemas operacionais com o código fonte fechado, distribuídos na forma binária apenas, como o Windows da Microsoft. Como na virtualização completa, os aplicativos tipicamente executam sem modificações. A Figura 1.4 ilustra a paravirtualização.

Grandes vantagens incluem performance, capacidade de alterar a escala e de gerenciamento. Os dois exemplos mais comuns desta estratégia são o Linux em modo Usuário (UML, User-mode Linux) e o Xen. A escolha da paravirtualização para o Xen tem se mostrado a que consegue obter maior desempenho e nível de isolamento mesmo num hardware de computador pessoal típico.

O Xen estende este modelo para E/S de dispositivos. Ele exporta interfaces de dispositivo simplificadas e genéricas para os sistemas operacionais hospedados. Isso é verdadeiro para um sistema Xen mesmo que ele utilize suporte de hardware para virtualização que permita aos sistemas operacionais executar sem modificações. Apenas os gerenciadores (drivers) para os dispositivos genéricos Xen precisam ser introduzidos no sistema.

FIGURA 1.4 Hipervisores de paravirtualização são semelhantes aos de virtualização completa, mas utilizam-se de versões modificadas dos sistemas operacionais hóspedes para otimizar a execução.

Virtualização em Nível de Sistema Operacional

Uma quarta técnica é *virtualização em nível de sistema operacional* (também chamada de *paenevirtualização* para refletir o fato de que ela é "quase virtualização"). Nessa técnica não existe monitor de máquina virtual. Em vez disso, tudo é feito inteiramente com uma única imagem tradicional de sistema operacional. Os SOs que suportam essa técnica são de propósito geral e compartilhamento de tempo com a capacidade de garantir fortemente espaços de nome e isolamento de recursos. Os

"hóspedes" criados em tais condições ainda são percebidos como se fossem máquinas separadas com seus próprios sistemas de arquivos, endereços IP e configurações de software. A Figura 1.5 ilustra a virtualização em nível de sistema operacional.

FIGURA 1.5 Com a virtualização em nível de sistema operacional, todos os servidores virtuais privativos são executados dentro do contexto de uma imagem de sistema operacional compartilhada e única destinada ao hardware físico subjacente da máquina hospedeira.

A vantagem da virtualização em nível de sistema operacional é que ela exige menos duplicação de recursos. Quando discutindo recursos em termos de virtualização de sistemas operacionais, a principal idéia por trás da arquitetura é exigir menos memória física para um sistema hóspede. Estes podem geralmente compartilhar algum espaço de programas de usuário, bibliotecas e mesmo conjuntos de softwares. No mínimo, cada uma dessas instâncias homogêneas de sistemas hóspedes não exige seu próprio núcleo de SO privado, porque todos seriam arquivos binários totalmente idênticos. Quando a virtualização em nível de sistema operacional é usada, os requisitos de memória para cada novo hóspede podem ser substancialmente reduzidos. Essa técnica é excepcional em situações que exijam extrema capacidade de mudança de escala e grande quantidade de hóspedes executando simultaneamente, já que uma quantidade maior destes pode ser contida numa dada quantidade de RAM física. Interessante notar que um hóspede aqui é algo substancialmente diferente dos discutidos até agora. Neste caso eles são contêineres de processos em espaços de usuário rigidamente isolados e associados e não uma instância completa de um sistema operacional, embora seja freqüente que se acredite que são.

Num ambiente onde cada hóspede deseja executar o mesmo sistema operacional, essa técnica pode ser adequada. No entanto, para muitos usuários, a capacidade de exe-

cutar diversos SOs na mesma máquina é o principal motivo de utilizar virtualização. Por definição a virtualização em nível de sistema operacional não concede essa vantagem.

Outra fraqueza da virtualização em nível de sistema operacional é que ela geralmente apresenta fraco isolamento entre os hóspedes. Conforme um deles vai consumindo mais recursos, a performance de todos os outros é prejudicada negativamente. Isso não é um problema fundamental. O sistema operacional subjacente pode ser alterado para conseguir isolamento forte, mas a experiência tem mostrado que a complexidade e o esforço necessário para atingir realmente esse objetivo são altos.

Em ambientes onde todos os hóspedes pertencem ao mesmo domínio administrativo, isolamento fraco pode ser aceitável porque os administradores podem ajustar a capacidade de alocar recursos para amenizar qualquer problema. Isso não pode ser feito em ambientes diversificados, onde múltiplos hóspedes são de proprietários diferentes e também administrados independentemente, operados sem nenhum incentivo específico para que colaborem eficientemente.

Implementações de virtualização em nível de sistema operacional incluem Virtuozzo, Linux VServers, OpenVZ, Solaris Containers, FreeBSD jails e HP UX 11i Secure Resource Partitions.

Outros Tipos de Virtualização

Dois tipos remanescentes de virtualização são dignos de mencionar, embora diferentes dos quatro tipos discutidos até agora, elas não são capazes de executar um sistema operacional completo. A primeira é a *virtualização de bibliotecas*, que emula sistemas operacionais ou subsistemas através de uma biblioteca de software especial. Um exemplo deste tipo é a biblioteca Wine disponível para sistemas Linux. O Wine possui um subconjunto da API do Win32 como uma biblioteca, para permitir que aplicativos Windows sejam executados em ambientes Linux.

O tipo final de virtualização que discutiremos neste capítulo é a *virtualização de aplicativos* (*tempo de execução gerenciado*). A virtualização de aplicativos é a abordagem de executar programas dentro de um ambiente virtual de execução. Isso é diferente de executar um aplicativo comum diretamente no hardware. O ambiente virtual de execução fornece uma API padronizada para execução entre plataformas e gerenciar o consumo de recursos locais utilizados. Ele também pode fornecer recursos como, por exemplo, o modelo de linhas de execução (threading model), variáveis de ambiente, bibliotecas de interface com os usuários e objetos que auxiliam na programação de aplicativos. O exemplo mais notável de tais ambientes virtuais de execução é a Máquina Virtual Java da Sun. É importante ter em mente que essa técnica não virtualiza todo o conjunto de hardware de um sistema operacional como um todo.

Visão Geral sobre os Tipos de Virtualização

A Tabela 1.1 contém um resumo sobre as técnicas de virtualização discutidas nesta seção.

TABELA 1.1 – Resumo sobre técnicas de virtualização

Tipo	Descrição	Vantagens	Desvantagens
Emuladores	O hipervisor fornece uma máquina virtual completa (de uma arquitetura de computação diferente da máquina hospedeira) que permite que aplicativos de outras arquiteturas executem no ambiente simulado.	Simula hardware que não está disponível fisicamente.	Baixa performance e densidade.
Completa	O hipervisor fornece uma máquina virtual completa (da mesma arquitetura de computação que o hospedeiro) que permite que hóspedes sem modificações executem isoladamente.	Flexibilidade – executar diferentes versões de múltiplos sistemas operacionais de diversas origens.	O Sistema Operacional hóspede não sabe que está sendo virtualizado. Pode ser causa de diminuição perceptível de performance num hardware comum, particularmente para aplicativos com intensivas operações de E/S.
Para	O hipervisor fornece uma máquina virtual completa, mas especializada (da mesma arquitetura de computação que o hospedeiro) para cada hóspede, que precisa ser modificado e executa em isolamento.	Leve e rápida, performance próxima da nativa: demonstrou-se que podem operar na faixa de 0,5% a 3% de processamento extra. `http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf` (em inglês). que o SO coopere com o hipervisor – melhora as E/S e alocação de recursos. Permite arquiteturas de virtualização que não suportam a forma completa. A Nível de SO	Exige que os sistemas operacionais sejam alterados para utilizar hiperchamadas em vez de alguns comandos críticos. A maior limitação da paravirtualização é o fato que o SO hóspede precisa ser personalizado para executar sobre o monitor da máquina virtual (VMM, Virtual Machine Monitor), que é o programa hospedeiro que permite que um único computador suporte diversos ambientes de execução idênticos. Isso tem um impacto significativo especialmente para sistemas operacionais antigos com código fechado que ainda não têmimplementadasextensões para paravirtualização.

Tipo	Descrição	Vantagens	Desvantagens
A Nível de SO	Um único sistema operacional é modificado de tal forma que permite vários processos servidores de espaço de usuário unidos em conjuntos funcionais, executados em isolamento mas ainda assim na mesma plataforma de hardware.	Camada de virtualização rápida e leve. Ela tem as melhores performance e densidade possíveis (ou seja, as mais próximas das nativas) e possui gerenciamento dinâmico de recursos.	Na prática, isolamento forte é difícil de implementar. Exige o mesmo sistema operacional, atualizado até exatamente o mesmo ponto no tempo, em todas as máquinas virtuais (infra-estrutura de computação homogênea).
De Bibliotecas	Emula sistemas operacionais ou subsistemas via uma biblioteca de software especial. Não dá a ilusão de sistema isolado com um sistema operacional completo.	Fornece APIs faltantes para desenvolvedores.	Com freqüência tem performance pior do que uma versão do aplicativo otimizada para a plataforma nativa em questão.
De Aplicativos	Aplicativos executam num ambiente virtual que fornece uma API padronizada para execução entre plataformas e gerencia o consumo local de recursos.	Gerencia recursos automaticamente, o que diminui a curva de aprendizado de programação. Aumenta a portabilidade das aplicações.	A execução é mais lenta do que a de código nativo. Processamento extra para manter a máquina virtual existe se comparando com a execução de código nativo.

O Legado da Virtualização

A virtualização é certamente uma parte importante dos ambientes de computação modernos, mas muitas pessoas não sabem que não é um conceito novo. De fato, as raízes da virtualização datam das origens da computação moderna. As implementações originais ainda existem em novas versões, mas são tipicamente utilizadas apenas em certos nichos como computadores de grande porte e de forma geral passaram despercebidas para a geração que testemunhou o surgimento da computação pessoal. O reaparecimento da virtualização nos tempos atuais apresentou o conceito para o público em geral. A Figura 1.6 apresenta uma linha do tempo com pontos-chave selecionados a respeito da virtualização. Nas próximas seções, falaremos sobre alguns desses pontos-chave para ajudar a dar um contexto histórico para o hipervisor Xen.

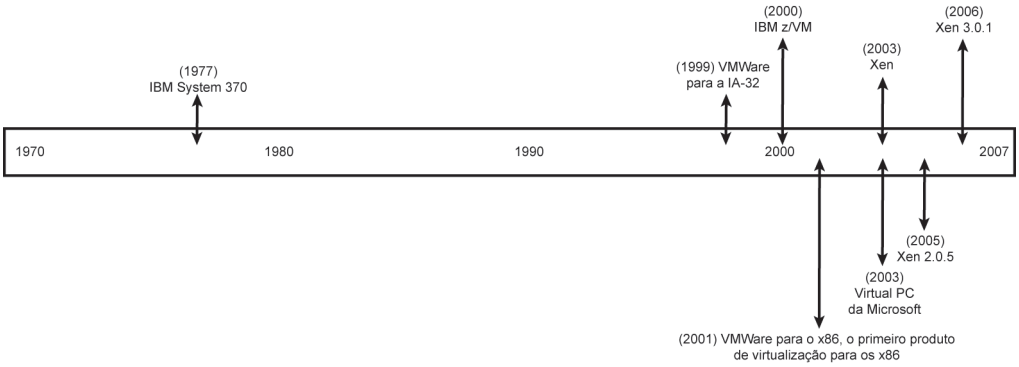

FIGURA 1.6 Linha do tempo mostrando alguns dos eventos mais importantes na história da virtualização

Os Computadores de Grande Porte IBM

Com o interesse comercial recente em virtualização para computadores mais comuns, é importante analisar as origens da tecnologia dos hipervisores. A virtualização se originou na década de 1960 em computadores de grande porte da IBM. Os primeiros pesquisadores em computação estavam interessados no aumento de robustez e estabilidade oferecidos pelos hipervisores. Eles permitiam que múltiplos sistemas operacionais hóspedes executassem simultaneamente e ainda assim garantiam que, se alguma das instâncias parasse de funcionar, as restantes ainda estariam isoladas. Com frequência, novos sistemas operacionais e áreas de pesquisa com código experimental eram desenvolvidos e depurados utilizando essas instâncias de produção estáveis.

O Sistema 370 da IBM foi o primeiro computador disponível comercialmente projetado para virtualização. Com a introdução do sistema operacional CP/CMS, múltiplas instâncias podiam ser executadas simultaneamente num hardware de grande porte Sistema 370 da IBM. A implementação de software era facilitada por um hardware tradutor de páginas que permitia suporte eficiente para memória virtual. A cooperação entre hardware e software para suportar a virtualização se tornou um dos principais atrativos da linhagem de computadores de grande porte da IBM. De fato, todos os computadores de grande porte da IBM da linha do Sistema z continuam a oferecer suporte para virtualização via hardware. O software que se aproveita da virtualização via hardware de forma mais completa é o z/VM, que atualmente superou o sistema original em CP/CMS. A abreviação VM (Virtual Machine, máquina virtual) indica que todas as interfaces de hardware são virtualizadas neste sistema operacional. O VM/CMS é muito bem avaliado e amplamente distribuído na indústria e em

ambientes acadêmicos. Muitas abordagens modernas de virtualização devem muito às implementações originais para computadores de grande porte da IBM.

Virtualização em Hardware Comum

Na década de 1990, o projeto Disco em Stanford liderado por Mendel Rosemblum usou máquinas virtuais para permitir que sistemas operacionais comuns executassem em hardware de computadores de acesso não uniforme à memória (NUMA, Non Uniform Memory Access). No entanto, neste caso o sistema operacional era o IRIX da Silicon Graphics, projetado para executar no MIPS R10000. Este processador, diferente do que acontecia com os computadores de grande porte da IBM, não foi projetado para dar suporte à virtualização completa. Em vez disso, os desenvolvedores do projeto Disco utilizaram uma técnica mais tarde conhecida como para-virtualização para introduzir modificações específicas para permitir a virtualização. Esses desenvolvedores modificaram e recompilaram o IRIX para permitir que ele executasse na arquitetura virtual modificada.

A equipe de Stanford voltou sua atenção para modificar outra plataforma comum não projetada para a virtualização, o x86. Isso teve uma influência direta na fundação da VMWare e a introdução do primeiro produto comercial de virtualização desta plataforma. Neste caso, conseguiram executar os arquivos binários sem alteração de sistemas operacionais como o Windows da Microsoft, efetuando tradução simultânea durante a execução para instruções não permitidas na arquitetura x86 modificada. Por exemplo, a instrução POPF (que retira dados da pilha e armazena o valor em registradores especiais para indicadores) precisa ser substituída porque quando executada em modo sem privilégios ela falha sem provocar erros e sem fazer as alterações requisitadas nos indicadores usados para desativar interrupções de hardware.

Extensões de Virtualização para o x86

Desde 2005, fabricantes de processadores como a Intel e a AMD aumentaram o suporte via hardware em suas linhas de produtos. A Tecnologia de Virtualização da Intel (VT, Virtualization Technology) foi desenvolvida utilizando o nome código de Vanderpool e a Virtualização da AMD (AMD-V) usando o nome código Pacífica. Esses hardwares promovem os objetivos da virtualização em plataformas comuns ao adicionar funcionalidades explícitas que permitem que hipervisores com performance melhor sejam utilizados com a técnica da virtualização completa, que fica tanto mais fácil de implementar quanto tem o potencial para melhor performance (lembre-se que o Xen utiliza essas extensões para dar suporte à virtualização completa).

As Origens do Xen e Sua Linha do Tempo

O hipervisor do Xen se originou no Laboratório de Computação da Universidade de Cambridge como parte do projeto XenoServer em andamento em 2001. Esse projeto tinha como objetivo criar uma infra-estrutura pública para computação distribuída por amplas áreas. A intenção do projeto era criar um sistema onde plataformas de execução do XenoServer estariam espalhadas pelo planeta para uso por qualquer membro no público-alvo. Quando a infra-estrutura do XenoServer estiver completa, seus usuários enviarão um código para ser executado e serão cobrados posteriormente pelos recursos utilizados durante a execução. Garantir que cada um dos nós físicos seja usado em sua máxima capacidade possível exige um hipervisor de alta performance capaz de hospedar múltiplos sistemas operacionais comuns num único servidor baseado em x86. Para cumprir tal papel, o Xen foi criado para ser o núcleo de cada nó do XenoServer. Ele permite estabelecer a responsabilidade pelo consumo de recursos, auditoria e, o mais importante, o gerenciamento de recursos necessário para a infra-estrutura do XenoServer. Para maiores informações a respeito do projeto, veja `http://www.xenoservers.net/`.

O Xen foi apresentado ao público pela primeira vez num artigo acadêmico aceito nos procedimentos de 2003 da Associação de Equipamentos de Computação (ACM, Association for Computing Machinery) para o Simpósio de Princípios em Sistemas Operacionais (SOSP, Symposium on Operating System Principles). A afirmação de ter a capacidade de executar virtualização rápida em máquinas x86 comuns criou um grande interesse na comunidade acadêmica. Essa afirmação foi verificada independentemente em ambientes acadêmicos, o que serviu para fortalecê-la. Em breve, um grande número de grupos se interessou por essa nova abordagem de virtualização. Nos anos seguintes a essa publicação inicial do Xen, muitas atualizações significativas ocorreram no projeto, permitindo melhorias na funcionalidade, confiabilidade e performance.

É digno de nota que, durante o desenvolvimento do Xen 1.x, a divisão de Pesquisas Microsoft, em colaboração com a Universidade de Cambridge, desenvolveu parte do Windows XP para o Xen. Essa adaptação se tornou possível em parte por causa do Programa de Licenciamento Acadêmico da Microsoft. Infelizmente, devido aos termos dessa licença, a adaptação nunca foi publicada, embora tenha sido mencionada no artigo original na SOSP sobre o Xen.

Uma empresa em separado, a XenSource, foi fundada em 2004 para promover a ampla adoção dos hipervisores de código aberto Xen pelo mundo empresarial. Essa empresa concentrou seus negócios em dar suporte ao desenvolvimento de

todo o núcleo de código aberto do Xen, enquanto ao mesmo tempo vendia para outras corporações pacotes e softwares de gerenciamento. Enquanto a XenSource liderava e coordenava os esforços de desenvolvimento, contribuições foram feitas por uma grande variedade de empresas e organizações incluindo IBM, Sun, HP, RedHat, Intel, AMD, SGI, Novell, a NSA, a Marinha dos EUA, Samsung, Fujitsu, Qlogic e muitas outras, incluindo nesse grupo pesquisadores de muitas universidades. Juntos, produziram um padrão em que todos podem confiar, que reduziu os riscos e acelerou o desenvolvimento para todos os participantes.

Durante o final do ano de 2004, o Xen 2.0 entrou em cena. A nova versão conseguiu grande flexibilidade na configuração de dispositivos virtuais de E/S dos sistemas operacionais hóspedes. Nesse ponto do desenvolvimento do Xen, os usuários podiam configurar regras de firewall arbitrárias, roteamento e pontes de interfaces hóspedes virtuais de rede. Suporte adicional foi acrescentado para volumes LVM de cópia-durante-gravação bem como para arquivos de retorno (loopback) para manter imagens de disco de sistemas operacionais hóspedes. O Xen 2.0 inclui suporte para multiprocessamento simétrico, embora imagens de hóspedes continuem com processadores únicos. A melhoria mais impressionante do ponto de vista da demonstração foi a adição da migração ativa (live migration), que permite que uma instância em execução de um sistema operacional seja movida entre hardwares conectados via rede sem interrupção perceptível no serviço. O Xen introduziu uma função de migração de hóspedes muito solicitada que impressionou muitos hackers casuais interessados em open source.

Durante o ano de 2005, uma comunidade cada vez maior de grupos interessados se formava em torno do Xen. Essa tendência era notável, tanto no mundo acadêmico quanto no empresarial, e no início de 2006 o Xen tinha conseguido uma participação significativa no total das virtualizações em uso. Grandes distribuidores de Linux ficaram cada vez mais interessados em se aproveitar da tecnologia do Xen para uso de seus clientes, e o suporte ao Xen se tornou padrão para muitos deles. O hipervisor do Xen 2.0 também suportava mais hóspedes do que antes, incluindo o Linux, o Open-Solaris, Plan 9 e muitas variantes do BSD.

Em 2006, o Xen 3.0 introduziu uma camada de abstração para tecnologias de virtualização de hardware oferecidas pelas implementações do Vanderpool da Intel e do Pacifica da AMD. Isso permitiu que hóspedes sem alteração (chamados de Máquinas Virtuais de Hardware, hóspedes HVM, Hardware Virtual Machines) além dos hóspedes para virtualizados tradicionais. O Xen 3.0 também incluiu suporte para sistemas hóspedes de multiprocessamento simétrico (SMP, Symmetric MultiProcessing) – incluindo CPUs virtuais de conexão dinâmica, suporte para grandes quantidades de memórias, para módulos de plataforma segura (TPM, Trusted Platform

Modules) e uma versão para a arquitetura IA64. Versões seguintes do Xen incluíam um novo agendador para CPUs com suporte para pesos, limites e balanceamento de carga SMP automática, bem como ferramentas de medida (Xen-oprofile), que permitiria aos desenvolvedores usuários de Xen otimizar código para obter ainda melhor desempenho no futuro. Outras características notáveis incluíam aumento na performance de rede graças a redução de carga de segmentação de pacotes, suporte melhorado para a IA64 e a primeira versão para a arquitetura de processadores Power.

Em 2007, a Citrix comprou a XenSource e esta se tornou o Grupo do Produto XenServer Citrix. Também em 2007, o Xen 3.1 foi lançado. Ele dava suporte à XenAPI, uma interface de programação para comandos Xen que permitia a integração de ferramentas de gerenciamento de terceiros, incluindo as baseadas no Modelo Comum de Informações da Força Tarefa de Gerenciamento Distribuído (DMTF CIM, Distributed Management Task Force's Common Information Model) que está se tornando um padrão para gerenciamento de agrupamentos de máquinas heterogêneas. Ela também possui capacidades de salvar/restaurar/migrar e controle dinâmico de memória para hóspedes HVM. A versão atual do Xen está disponível em `http://xen.org/download/`. Quando este livro estava a ponto de ser publicado, a versão liberada mais recente do Xen era a 3.2.

Outros Sistemas de Virtualização Para Hardware Comum

O Xen não é o único sistema de virtualização disponível, mas tem uma combinação de características que o tornam adequado de forma única para muitos aplicativos importantes. Ele executa em plataformas comuns. É feito com código aberto. É rápido, mostrando performance próxima da nativa. Pode-se facilmente mudar a escala de projetos feitos com ele. Ele oferece características de servidor como migração ativa. Suporta hóspedes sem modificações como o Windows, mas é também capaz de paravirtualização. Muitas empresas estão se unindo para utilizar o Xen como uma plataforma de código aberto para a qual podem contribuir com a garantia que seus esforços não ficarão ocultos por trás de uma plataforma proprietária que pode mudar inesperadamente.

Uma última característica de virtualização específica que discutiremos é o tipo de instalação. Existem dois tipos: *hospedada* e *direta* (em inglês, *bare-metal*, literalmente "direto no metal"). Hospedada quer dizer que o sistema de virtualização é instalado e executa num sistema operacional hospedeiro. Direta implica que o sistema de virtualização é instalado e executa diretamente no hardware físico, e um sistema operacional hospedeiro não é necessário. O Xen é um sistema de virtualização direto. Lembre-se também que ele suporta paravirtualização e virtualização completa.

Esta seção discute uma variedade de tecnologias de virtualização comuns no mercado hoje em dia. Antes discutimos diferentes tipos de virtualização de um ponto de vista abstrato. Aqui veremos exemplos concretos de implementações de cada abordagem bem como suas vantagens e desvantagens. Se você não está interessado nessas outras tecnologias de virtualização, pode pular essa seção. Não se preocupe, não existem perguntas a respeito deste tópico mais adiante.

Emuladores

- **Bochs** – um simulador de computador x86 para executar numa variedade de plataformas, incluindo o x86, PowerPC, SPARC, Alpha e MIPS. O Bochs pode ser configurado para emular muitas gerações da arquitetura de computação x86 incluindo 386, 486, Pentium, Pentium Pro e implementações mais modernas de 64 bits. Adicionalmente instruções adicionais como MMX, SSE, SSE2 e 3DNow podem ser também emuladas. O Bochs distingue-se por simular não somente o processador, mas também o sistema inteiro, incluindo os periféricos necessários para operação normal. Estes incluem o mouse, teclado, hardware para gráficos e de rede. O Bochs é capaz de executar diversos sistemas operacionais como hóspedes, incluindo muitas edições do Windows, entre elas o XP e o Vista, além do DOS e do Linux. É importante notar que o Bochs precisa de um sistema operacional hospedeiro para funcionar e não é capaz de instalação direta. Ele é tipicamente hospedado em Linux, Windows e Mac OS X.
- **QEMU** – outro exemplo de emulador, mas é interessante notar as formas pelas quais ele difere do Bochs. O QEMU oferece dois modos de operação. O primeiro é o modo de simulação de sistema completo, que é semelhante ao Bochs já que um computador pessoal inteiro, com periféricos, é emulado. Neste modo, uma série de arquiteturas de processadores pode ser simulada, como a x86, x86_64, ARM, SPARC, PowerPC e MIPS, com velocidade razoável utilizando tradução dinâmica. Usando este modo, pode-se emular um ambiente capaz de hospedar os sistemas operacionais Windows da Microsoft (incluindo o XP) e Linux e pode ser hospedado em Linux, Solaris e FreeBSD.

Combinações adicionais de sistemas operacionais são também suportadas. O segundo modo de operação é conhecido como Emulador de Modo de Usuário. Esse modo só está disponível quando se hospeda em Linux e permite que arquivos binários de uma arquitetura diferentes sejam executados. Por exemplo, arquivos binários compilados para a arquitetura MIPS podem ser executados num Linux rodando numa arquitetura x86. Outras suportadas nesse modo incluem SPARC,

PowerPC e ARM, e mais estão em desenvolvimento. O Xen se aproveita do modelo de dispositivos do QEMU para hóspedes HVM.

Virtualização Completa

- **VMWare –** Fundada em 1998, foi a primeira empresa a oferecer comercialmente softwares para virtualização para o x86 e atualmente possui uma extensa linha de produtos para soluções de virtualização para servidores e máquinas pessoais baseadas nesta arquitetura. A VMWare tem um produto de instalação direta, o ESX Server. Nele, um hipervisor fica entre os sistemas operacionais hóspedes e o hardware em si como uma camada de abstração. Com a Estação de Trabalho (Workstation) VMWare, o hipervisor executa hospedado como um aplicativo instalado sobre um sistema operacional base como o Windows ou Linux. É possível executar sistemas operacionais sem modificação utilizando tradução simultânea com a execução de instruções que não podem ser virtualizadas. O Servidor ESX e a Estação de Trabalho são produtos comerciais. Existe uma versão do servidor que é hospedada que pode ser baixada gratuitamente, bem como o VMWare Player, que permite que usuários executem máquinas virtuais criadas pelo Servidor ou pela Estação de Trabalho. A VMWare também mantém uma extensa biblioteca de aplicativos para máquinas virtuais pré-configurados e gratuitos. Veja em `http://www.vmware.com/appliances/`. Note que é possível executar hóspedes paravirtualizados com a VMI da VMWare (descrita na seção paravirt_ops mais adiante neste capítulo).
- **Microsoft –** tendo entrado relativamente tarde no mercado comercial de virtualização, a empresa está para lançar o Hyper-V, um hipervisor que estará disponível tanto como um produto isolado quanto como uma característica do Windows Server 2008. A Microsoft tem também um produto que precisa executar hospedado que está disponível gratuitamente para o Windows chamado Virtual PC. Ele foi inicialmente implementado para os Macintosh da Apple por uma empresa chamada Connectix. A primeira versão foi lançada no meio de 1997. Em 2001, um suporte para Windows foi lançado com o nome Virtual PC para Windows 4.0. Dois anos mais tarde, em 2003, a Microsoft percebeu que virtualização estava se tornando cada vez mais importante para o mercado empresarial e então comprou a Virtual PC e um produto ainda não lançado chamado Virtual Server. Esse software está atualmente no Mercado como Microsoft Virtual PC 2007. Existe também uma versão do Virtual PC para Mac disponível comercialmente. O Virtual PC 2004 emula uma CPU Intel Pentium 4 com um chipset Intel 440BX (esse chipset suporta processadores Pentium II,

III e Celeron), uma placa de vídeo padrão SVGA Vesa, uma placa de som e um dispositivo Ethernet. Desde então, a Microsoft lançou a versão para Windows para ser baixada gratuitamente, enquanto que a para Mac continua disponível comercialmente. A edição de 2007 suporta o Windows Vista como hóspede (apenas a versão de 32 bits e sem suporte para a nova interface Aero Glass) e como hospedeiro (tanto 32 bits quanto 64). Atualmente o Virtual PC 2007 está disponível apenas para Windows. É digno de nota mencionar que as implementações para ambas as plataformas usam a técnica de recompilar dinamicamente para realizar seu trabalho. Edições anteriores do Virtual PC que executavam no Macintosh utilizavam esta técnica para converter instruções x86 desconhecidas na plataforma para as nativas do PowerPC. Interessante notar que as edições do Virtual PC para Windows empregam a mesma técnica por um motivo completamente diferente. A edição para Windows traduz código x86 hóspede em modo kernel e em modo real para código x86 em modo de usuário. Qualquer código hóspede neste modo executa nativamente. Chamadas nos hóspedes são capturadas em algumas circunstâncias para melhorar a performance ou a integração com o ambiente que hospeda o produto.

- **Linux KVM (Kernel Virtual Machine, Ncleo Mquina Virtual) –** outro iniciante no mundo da virtualização é a Linux KVM. Ela é uma solução de virtualização completa formada durante o período do desenvolvimento do kernel (núcleo funcional do SO) principal 2.6.20. A KVM é uma modificação do núcleo do Linux que efetivamente o transforma num hipervisor quando se acrescenta um módulo adicional. O método pelo qual a KVM opera é muito interessante. Cada hóspede na KVM é na realidade executado no espaço de usuários do sistema hospedeiro. Esta abordagem faz com que cada instância hóspede (um dado kernel hóspede e seu espaço de usuários associado) pareça como um processo normal para o kernel hospedeiro subjacente. Assim a KVM tem isolamento mais fraco do que outras abordagens discutidas até agora. Com a KVM, o agendador de processos bem ajustado do Linux efetua as tarefas de um hipervisor, multiplexando as máquinas virtuais da mesma forma que faria com processos de usuário em operação normal. Para conseguir isso, a KVM introduz um novo modo de execução distinto dos modos tradicionalmente encontrados num sistema Linux (kernel e usuário). Este novo modo projetado para hóspedes virtualizados é chamado de forma bastante adequada de modo hóspede (guest mode). Este modo tem seus próprios modos de usuário e de kernel. Ele é usado quando executando todo o código hóspede que não seja de E/S, e a KVM cai de volta para modo de usuário normal para dar suporte a operações de E/S para hóspedes virtuais. O modulo KVM emprega um controlador (driver) de dispositivo com novas carac-

terísticas para expor hardware virtual para as instâncias hóspedes através da entrada /dev/kvm no sistema de arquivos do hipervisor. Os hóspedes KVM então acessam o hardware virtual via um processo QEMU ligeiramente modificado. Quando executa em hardware atual com extensões de virtualização, hóspedes Linux (32 bits e 64) e Windows (32 bits) são possíveis. Embora o suporte de hardware seja necessário para hóspedes Windows, a KVM está sendo desenvolvida para se aproveitar da paravirt_ops para hóspedes Linux.

Paravirtualização

- **Linux em modo usurio –** a implementação de paravirtualização conhecida como *Linux em modo usuário* (*UML, User-Mode Linux*) permite que este sistema operacional execute outros sistemas Linux em espaço de usuário. Cada instância de máquina virtual executa um processo no Linux hospedeiro. O UML é especialmente projetado para executar VMs de hóspedes Linux num sistema Linux. O UML foi consolidado durante o período de desenvolvimento do kernel 2.6 do Linux. Os hóspedes UML possuem como complemente os tipos usuais de dispositivos virtuais encontrados nas implementações de virtualização da atualidade. Porque o UML é uma forma de paravirtualização, os kernels hóspedes precisam ter certas opções ativas em seus arquivos de configuração antes de serem compilados. Os kernels UML podem ser aninhados, permitindo pilhas de máquinas virtuais executando umas dentro das outras com uma única instância hospedeira do Linux como base. A maior desvantagem do UML é que ela é uma VM destinada a ser uma solução exclusivamente para Linux. Embora seja razoável conceber tal projeto, esta estratégia talvez não seja adequada para os ambientes de computação heterogêneos da atualidade que precisam das vantagens oferecidas pela virtualização.
- **Lguest –** um outro método de virtualização encontrado na linha principal do kernel do Linux. Ele é mantido por Rusty Russel e foi consolidado durante o período de desenvolvimento do kernel 2.6.23. Um detalhe muito interessante sobre o lguest é que ele é implementado como um módulo do kernel, diferente de outros métodos de virtualização que se utilizam de mecanismos muito diferentes, descritos neste livro. Embora ele possa não ser tão funcional quanto algumas das outras suas contrapartes, é uma excelente ferramenta para aprendizado e experimentação de implementações para virtualização devido à sua quantidade de código relativamente pequena. Uma versão experimental de 64 bits do lguest está em desenvolvimento atualmente na Red Hat. Embora o lguest seja novo, sua inclusão rápida nas fontes dos kernels

de primeira linha o faz digno de menção. Você pode aprender mais sobre ele na página do projeto: `http://lguest.ozlabs.org/`.

- **Uma interface de paravirtualizao comum: paravirt_ops –** o histórico da virtualização diretamente no kernel possui registrado alguns desentendimentos logo no início do processo a respeito de como melhor implementar a técnica. Não existia nenhum padrão para a interface no kernel para implementar paravirtualização. Enquanto o Xen utilizava uma abordagem, a VMWare propôs uma ABI alternativa que funcionava em diversas plataformas chamada Interface de Máquina Virtual (VMI, Virtual Machine Interface) em 2005. A VMI foi consolidada dentro do kernel durante o período de desenvolvimento da versão 2.6.21. Enquanto isso, outras implementações estavam empregando suas próprias abordagens distintas. Os desenvolvedores do kernel do Linux perceberam que algum tipo de solução de implementação teria de dar suporte às necessidades de todos os desenvolvedores para ser boa. Assim eles empregaram uma abordagem encontrada em outras partes do código do kernel do Linux: com freqüência, quando se antecipa que existirão diferentes implementações de uma dada função numa API, uma estrutura de operações é utilizada para manter aquela API individual numa forma coerente. A implementação proposta é conhecida como estrutura de operações de paravirtualização (também conhecida como estrutura paravirt_ops*). Este método não obriga que nenhuma ABI em particular exista, permitindo em vez disso seleção em tempo de execução da implementação efetiva de métodos na API. Cada plataforma de virtualização pode implementar suas próprias funções específicas para essa interface comum. O Linux consolidou pacotes de atualização para dar suporte à paravirt_ops durante o ciclo de desenvolvimento da linha principal de kernel de versão 2.6.23. Essa padronização recente afetando as capacidades de virtualização do Linux ocorreu graças à cooperação de muitos desenvolvedores envolvidos em virtualização e deve facilitar o desenvolvimento mais rápido de aplicativos que se utilizem das tecnologias respectivas. Com essa infra-estrutura, um kernel que suporte a paravirt_ops deve ser capaz de executar nativamente, como um hóspede do Xen, ou como um hóspede utilizando a VMI.

* (NT): onde "ops" parece ser abreviação de operations structure, estrutura de operações.

Virtualização em Sistema Operacional

- **Linux-VServer –** é um exemplo de virtualização em sistema operacional para hardware comum. Suportada tanto pelo kernel Linux 2.4 quanto pelo 2.6 e operando numa variedade de plataformas de hardware (x86, x86-64, SPARC, Po-

werPC, MIPS, ARM), ele virtualiza uma única instância do kernel do Linux de forma que múltiplos hóspedes possam ser instanciados em espaço de usuário. Na terminologia do VServer, os hóspedes são chamados de servidores privativos virtuais (*VPS, Virtual Private Servers*). Os VPSs executam independentemente sem conhecimento a respeito da existência dos demais. Medidas extras são tomadas para garantir que tais processos VPS fiquem isolados – isso feito via modificações nas estruturas de dados e algoritmos internos do kernel do Linux. Os Linux-VServers se aproveitam de uma variante estrita do chroot para isolar o diretório raiz de cada VPS de forma que é impossível que se escape dessa raiz. Dessa forma, o VPS é isolado à sua própria parcela do sistema de arquivos e não pode examinar o que seu hospedeiro ou o que outros VPSs estão fazendo. As modificações no VServer introduziram também um contexto de execução, que é conceitualmente um contêiner que une grupos de processos VPS. Através de um contexto, ferramentas como top ou ps mostram apenas resultados relevantes para um único VPS. Quando o sistema é inicializado, o kernel define um contexto padrão para sua própria execução. Além disso, os VServer utilizam um contexto expectador especial para tarefas administrativas como visualizar todos os processos no sistema independentemente de qual contexto o contém.

O agendamento de execução de processos no VServer é conseguido utilizando-se uma metáfora de recipiente (bucket, literalmente "balde"). Um recipiente de capacidade determinada é preenchido com uma quantidade de unidades de execução (tokens) a cada certo intervalo especificado até que o recipiente esteja cheio. Note que qualquer unidade em excesso é "derramada" pra fora do recipiente. A cada ciclo de temporização, um processo que efetivamente precise (e se utilize) da CPU consome exatamente uma das unidades do recipiente, a não ser que ele esteja vazio. Uma vez que o recipiente se esvazie, o processo é colocado numa fila de espera até que o recipiente tenha sido preenchido de novo com uma quantidade mínima de M unidades, e nesse ponto o processo é agendado para executar novamente. Esse projeto de agendador oferece vantagens porque unidades podem ser acumuladas quando a máquina estiver mais livre para serem utilizados quando necessários mais tarde. Tal recipiente de unidades de execução em contextos permite controle detalhado sobre a utilização de processamento de todos os processos confinados. Uma modificação dessa abordagem como implementada nos VServers é de modificar dinamicamente valores de prioridade baseando-se no nível atual de preenchimento do recipiente.

- **OpenVZ** – outra implementação de virtualização via sistema operacional, similar em essência aos Linux-VServer, mas distinta. Esta implementação também funciona numa variedade de arquiteturas de hardware, incluindo o x86, x86-64 e PowerPC. O OpenVZ é implementado como um kernel modificado que su-

porta seu próprio ambiente virtual e espaço de usuários isolado conhecido como VE*. Os VEs são também chamados de VPSs, a mesma sigla e significado do Linux-VServer. Os recursos utilizados pelos VEs são controlados por construtos chamados de *beancounters* (literalmente "contadores de feijão"). Eles são constituídos de uma série de parâmetros que definem a distribuição de recursos para a instância do VPS. Eles definem a memória de sistema disponível para a instância, o número total de objetos IPC disponíveis e outros parâmetros de sistema. Uma diferença de implementação digna de nota é o agendador de execução de processos de ação dual. Esse agendador único primeiro seleciona uma instância de VPS da lista de servidores virtuais e então invoca um segundo nível de agendamento para escolher qual dos processos do espaço de usuário hóspede deve ser executado (e respeitando os indicadores padrão de prioridade de processo do Linux). O OpenVZ também oferece pontos de checagem de sistema operacional hóspede. Um ponto de checagem é criado quando um hóspede é impedido de iniciar novos processos e tem seu estado "congelado" e serializado para um arquivo. Um arquivo desses pode então ser transportado para outro OpenVZ e ativado. Infelizmente, como em muitas soluções comuns de virtualização, o OpenVZ vem com outro conjunto específico implementado de ferramentas de administração para gerenciamento. Como uma nota final, o OpenVZ suporta a migração ativa que muitos administradores de sistema acham útil.

Produtos de Virtualização Populares

A Tabela 1.2 contém uma lista de produtos de virtualização populares junto com informaçõessobre seu tipo e licença. Embora existam diferenças importantes e sutis entre elas, as licenças GPL (GNU General Public Licence, Licença Geral Pública), a LGPL (GNU Lesser General Public Licence, Licença Pública Menos que Geral) e a CDDL (Common Development and Distribution Licence, Licença de Desenvolvimento e Distribuição Comum) são todas tipos de licenças de código aberto. As licenças BSD e MIT são licenças bem liberais escritas pela Universidade da Califórnia, em Berkeley, e pelo Instituto de Tecnologia de Massachusetts, respectivamente. Como outras licenças de código aberto, elas permitem o uso do código fonte e na realidade impõem menos restrições de uso do que as licenças no estilo GPL. Para maiores informações sobre as licenças de código aberto veja: `http://www.opensource.org/licences`.

* (NT): onde VE deve ser sigla de virtual environment, ambiente virtual.

TABELA 1.2 – Visão Geral sobre Produtos de Virtualização

Implementação	Tipo de Virtualização	Tipo de Instalação	Licença
Bochs	Emulador	Hospedado	LGPL
QEMU	Emulador	Hospedado	LGPL/GPL
VMWare	Completa e Paravirtualização	Hospedada e Direta	Proprietária
Linux Modo Usuário (UML)	Paravirtualização	Hospedado	GPL
Lguest	Paravirtualização	Direta	GPL
Open VZ	Nível de SO	Direta	GPL
Linux VServer	Nível de Sistema Operacional	Direta	GPL
Xen	Paravirtualização, ou Completa quando utilizadas extensões de hardware	Direta	GPL
Parallels	Completa	Hospedada	Proprietária
Microsoft	Completa	Hospedada	Proprietária
z/VM	Completa	Hospedada e Direta	Proprietária
KVM	Completa	Direta	GPL
Contêineres Solaris	Nível de SO	Hospedada	CDDL
Gaiolas BSD (Jails)	Nível de SO	Hospedada	BSD
Mono	Nível de aplicativos	Camada de aplicativos	Compilador e ferramentas, GPL, bibliotecas de tempo de execução LGPL, bibliotecas de classes MIT X11
Máquina Virtual Java	Nível de aplicativos	Camada de aplicativos	GPL

Resumo

Neste capítulo, apresentamos os conceitos básicos por trás da virtualização, bem como os benefícios que se pode obter com ela. Esperamos ter dado uma base sólida ao apresentar os vários tipos de virtualização existentes hoje em dia, para que você possa comparar possíveis estratégias de virtualização baseando-se nos tipos implementados. Examinamos os precursores históricos do Xen começando com hardware especializado na década de 1960 e discutimos a evolução do hardware que permite que virtualizadores como o Xen operem em hardware atual comum. Entender este material serve como pré-requisito para ter a capacidade de fazer comparações fundamentadas entre o Xen e outras tecnologias. Porque nesta discussão optamos por escolher o Xen como plataforma de virtualização, no restante deste livro nos concentraremos exclusivamente nele.

Para Referência e Mais Informações

"x86 Virtualization." Wikipedia.
http://en.wikipedia.org/wiki/Virtualization_Technology.
"Comparison of Virtual Machines." (Comparativo entre máquinas virtuais) Wikipedia.
http://en.wikipedia.org/wiki/Comparison_of_virtual_machines.
"Emulation." (emulação) Wikipedia.
http://en.wikipedia.org/wiki/Emulation.
"Full Virtualization." (Virtualização Completa) Wikipedia.
http://en.wikipedia.org/wiki/Full_virtualization.
"Popek and Goldberg Virtualization Requirements." (Requisitos de Popek e Goldberg para Virtualização) Wikipedia.
http://en.wikipedia.org/wiki/Popek_and_Goldberg_virtualization_requirements.
"Xen." Wikipedia.
http://en.wikipedia.org/wiki/Xen.
"Xen." Xen.org.
http://xen.org/xen/.
"The Xen Virtual Machine Monitor." (O Monitor de Máquina Virtual Xen) University of Cambridge.
http://www.cl.cam.ac.uk/research/srg/netos/xen/.
Xen Project Status and Roadmap from the 2007 Xen Summit. (Situação do Projeto Xen e Orientações Originadas no Encontro de 2007 Sobre o Xen)
http://xen.org/xensummit/
http://community.citrix.com/blogs/citrite/barryf/2007/12/14/Xen+Project+Status+and+Roadmap+from+Xen+Summit
"Relationship between Xen Paravitualization and Microsoft Enlightenment." (Relação entre a Paravirtualização Xen e a Iluminação da Microsoft)
http://servervirtualization.blogs.techtarget.com/2007/03/30/steps-along-the-path-of-enlightenment/
"virtualization.info: News digest and insights about virtual machines and virtualization technologies, products, market trends since 2003." (resumo de notícias e opiniões sobre máquinas virtuais e tecnologias de virtualização, produtos e tendências de mercado desde 2003)
http://www.virtualization.info/