VIRTUALIZAÇÃO
O HARDWARE VIRTUAL

Autor: Me. Ricardo César Ribeiro dos Santos

Revisor: Luciana de Castro Lugli

INICIAR

introdução

Introdução

Quando se trata do contexto de virtualização, é importante considerar o hardware básico emulado pelo ambiente, composto pelo disco rígido virtual, o processador virtual e a memória virtual, sem considerar os outros dispositivos de entrada e saída. No contexto de virtualização, cada máquina virtual tem acesso a uma réplica de *hardware* que é nada mais que a abstração do sistema hospedeiro. Este modelo frequentemente tem características diversas à sua contraparte real, seja por que os drivers do hypervisor não conseguem replicar funções realizadas por alguns dispositivos, seja por decisão explícita do administrador do sistema, que decide provisionar desempenho para melhor atender a todos os recursos necessários.

Nesta unidade, serão apresentados os principais conceitos que influenciam a escolha do *hardware* virtual, formas de criar e acessar os dispositivos, os algoritmos que regem o funcionamento dos dispositivos e uma introdução a técnicas de otimização.

Processadores Virtuais

Segundo Portnoy (2012), uma das tarefas do hypervisor é o escalonamento das máquinas virtuais. Neste contexto, o hypervisor tem a tarefa de controlar o acesso de uma máquina virtual ao processador do computador em que está sendo executada da mesma forma que o sistema operacional que realiza este controle com aplicativos.

Para entender o escalonamento, é necessário entender como funciona a divisão de tempo de tarefas. Considere a situação mostrada na Figura 3.1.

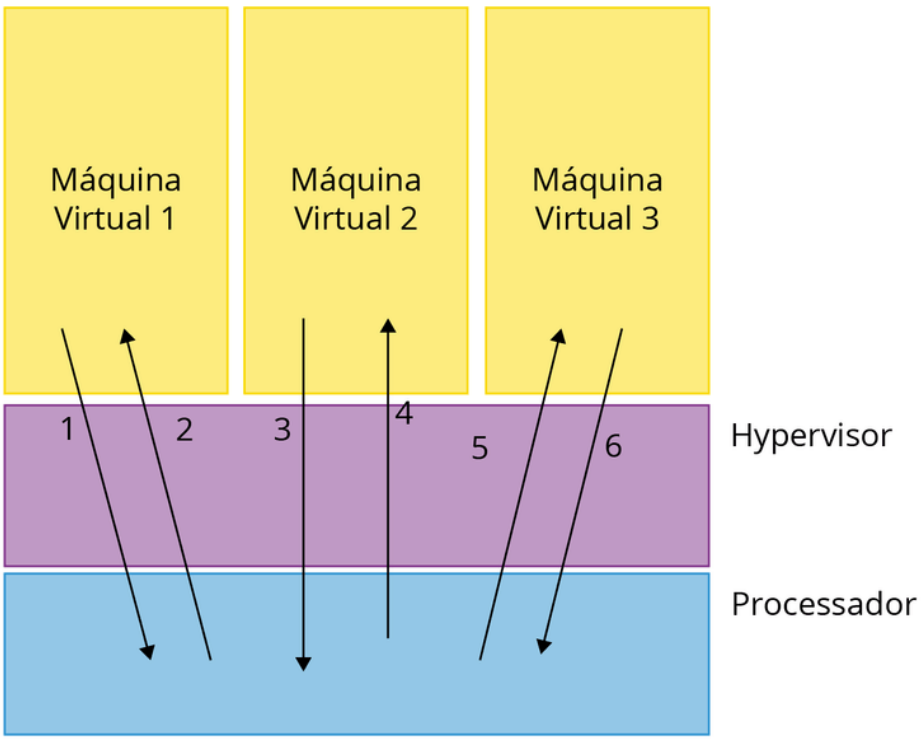


Figura 3.1 - Três máquinas virtuais executando através de um hypervisor em um computador hospedeiro
Fonte: Adaptada de Portnoy (2012, p. 127).

Na Figura 3.1 existem seis requisições sendo feitas para o processador do sistema hospedeiro e que serão atendidas. Cada requisição é composta por um conjunto de códigos correspondentes a sequências de instruções do microprocessador. Cada instrução é atômica (ou seja, indivisível) e o resultado deve ser retornado ao processo que fez a requisição.

Somente seria possível que todas as requisições da Figura 3.1 fossem atendidas ao mesmo tempo se o processador tivesse pelo menos três núcleos. Atualmente, é comum os microprocessadores terem 4 núcleos de processamento ou mais, isso viabiliza a realização da operação desta forma.

Quando se cria uma máquina virtual, entretanto, é necessário especificar que tipo de processador virtual ela terá. Caso se configure a máquina virtual com um processador virtual de dois núcleos, dois núcleos do processador deverão estar disponíveis para essas operações, por exemplo.

Nesse contexto, é possível notar que é possível que um gargalo comece a se formar: O processador precisará lidar com mais requisições para que seja

possível realizar as operações demandadas, o que pode ocasionar uma falta de núcleos de processamento livres para todas as máquinas virtuais.

A solução para este problema é o escalonamento de atividades. Esta técnica cria fatias de tempo em que atividades podem ser alocada. Por exemplo, na Figura 3.1 é possível que o processador execute as instruções 1, 2, 3, 4, ,5 e 6 cada uma em uma fatia de tempo diferente.

É necessário notar que não necessariamente as funções serão executadas nesta sequência, uma vez que a forma de escalonamento das operações é realizada pelo processador conforme seu algoritmo interno.

Dessa forma, alternando rapidamente entre tarefas, dá a impressão ao usuário de que todas as máquinas virtuais estão executando ao mesmo tempo, quando, na verdade, estão dividindo o processador entre si.

Porém, é necessário notar que o processo de escalonamento descrito até aqui não é o único que acontece. Além do escalonamento das máquinas virtuais, realizado pelo hypervisor, as máquinas virtuais também escalonam, internamente, o acesso de seus processos aos seus processadores virtuais, introduzindo *overhead* na operação.

Saiba mais

O termo *Overhead* tem origem na contabilidade e é utilizado para denotar todas as despesas e custos com que uma empresa arca ligados ao funcionamento, independentemente do produto ou serviço produzido. Em informática, o termo é utilizado para designar um custo computacional que é independente do serviço prestado ao usuário, ou seja, um custo computacional que não está atrelado à execução das aplicações que o usuário deseja executar. Podem ser os metadados de um arquivo em disco, as informações de endereçamento em uma aplicação de redes ou o tempo de espera necessário para os processos utilizarem o processador.

Fonte: Unionpédia Comunicação (2019).

Considerando-se o conjunto das operações de escalonamento do sistema operacional hóspede, instalado nas máquinas virtuais, e do hypervisor escalonando as máquinas virtuais propriamente ditas, é possível notar que o overhead em máquinas virtuais é maior que em suas versões não virtualizadas. Este custo computacional, entretanto, é inerente ao funcionamento do modelo e não pode ser eliminado.

Porém, é possível minimizar este custo quando existem mais de uma máquina virtual executando no mesmo hospedeiro. Para verificar como isso é possível, é necessário relembrar que um processador virtual é simplesmente uma

abstração do processador real.

Ou seja, o processador virtual (a vCPU) pode ter uma quantidade de núcleos diferente da do processador real. E, apesar de ser possível emular um processador virtual com até mais núcleos que o processador real, pode ser interessante criar um sistema virtual com menos núcleos.

É importante notar que os microprocessadores atuais possuem, em geral, mais de um núcleo, o que quer dizer que podem executar mais de uma tarefa ao mesmo tempo. Desta forma, controlando a quantidade de núcleos dos microprocessadores virtuais, um administrador pode otimizar a execução simultânea de mais de uma máquina virtual. Um exemplo de como o administrador pode realizar este controle é ilustrada na Figura 3.2.

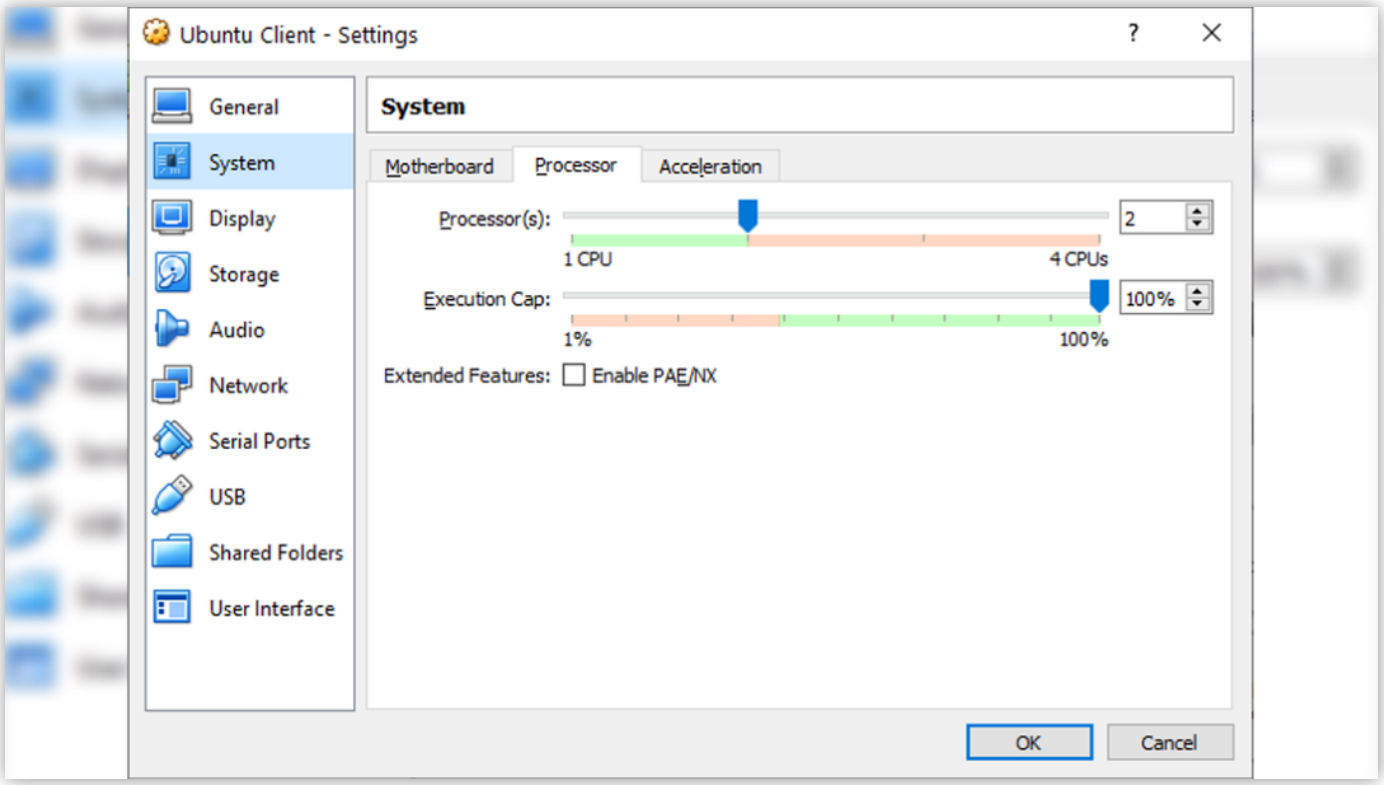


Figura 3.2 - Configurações do processador virtual no software VirtualBox, executando em um microprocessador Intel i7
Fonte: Elaborada pelo autor.

A Figura 3.2 mostra as configurações do processador de uma máquina virtual instalada no *software* VirtualBox. Como está executando em um sistema que possui um processador Intel i7, que conta com quatro núcleos, é possível configurar a máquina virtual para utilizar toda a máquina, mas o hypervisor

recomenda utilizar apenas dois dos quatro núcleos.

Expandindo um pouco sobre o exemplo anterior, caso duas instâncias desta máquina estivessem executando, deveriam ser escalonadas através de todos os núcleos do sistema hospedeiro.

Finalmente, é importante notar que não há o recurso para o administrador do sistema controlar qual núcleo deverá ser alocado para qual máquina virtual, isso é realizado de maneira automatizada pelo hypervisor.

praticar

Vamos Praticar

Juntamente com memória, I/O de rede e I/O de armazenamento, CPU é um dos recursos principais para determinar quão bem um servidor está se comportando. Uma das práticas centrais de virtualização [...] é que deve ter pouca ou nenhuma diferença de performance entre um sistema virtualizado e seu correspondente físico. Se um dos recursos sofre contenção ou está restringido, toda a performance do servidor virtual parece degradado, ainda que o gargalo seja somente um dos componentes (PORTNOY, 2012, p.125, tradução nossa).

Em relação ao assunto assinale a alternativa **correta** :

- **a)** O processador virtualizado deve ser exatamente igual ao do sistema hospedeiro. Não é permitido modificar a configuração de nenhuma maneira.
- **b)** O hypervisor deve escalonar o acesso das máquinas virtuais à CPU do sistema hospedeiro e a cada máquina virtual é designado uma parte do processamento.

- **c)** A própria máquina virtual deve ceder o controle da CPU toda vez que possível, de forma a garantir tempo de CPU para todas as máquinas virtuais.
- **d)** No momento da criação de máquinas virtuais, o processador virtual não é definido. O próprio hypervisor verifica as necessidades da máquina criada.
- **e)** A máquina virtual não tem processador declarado. O hypervisor sempre assume que a configuração é similar à da CPU do sistema hospedeiro.

Sistemas de Arquivos - Discos Virtuais

Assim como o processador virtual, a máquina virtual também possui um disco rígido virtual. E, assim como todo processamento, o processo de acesso e recuperação de informações também é similar ao de um computador real.

Quando um processo determina que necessita de uma informação que não tem disponível, ele envia uma requisição de informação para o sistema operacional, que deve processar este pedido e, utilizando seus drivers, busca no armazenamento do computador o dado requisitado.

Existe um outro cenário em que o disco rígido não é interno ao computador, o armazenamento é remoto ao ambiente que está executando na máquina. Neste caso, trata-se de um sistema de armazenamento na rede, em que uma requisição deve ser feita ao driver de rede que se encarregará de pesquisar o dado remotamente no computador que o usuário está utilizando.

Em ambos os casos, o sistema operacional se encarrega de buscar os dados e disponibilizar para o aplicativo que os requisitou.

No caso de um sistema virtualizado, o procedimento se mantém o mesmo para o sistema hóspede. As requisições ainda são realizadas para os drivers de disco rígido ou de rede dependendo do caso.

A diferença, entretanto, é que ao invés da requisição ser encaminhada diretamente para o hardware, ela vai para o hardware virtual que o hypervisor cria para esta máquina virtual. Então, o pedido é encaminhado para o driver apropriado instalado no hypervisor e é só então que o pedido é direcionado para a interface de hardware apropriada. A Figura 3.3 ilustra o processo.

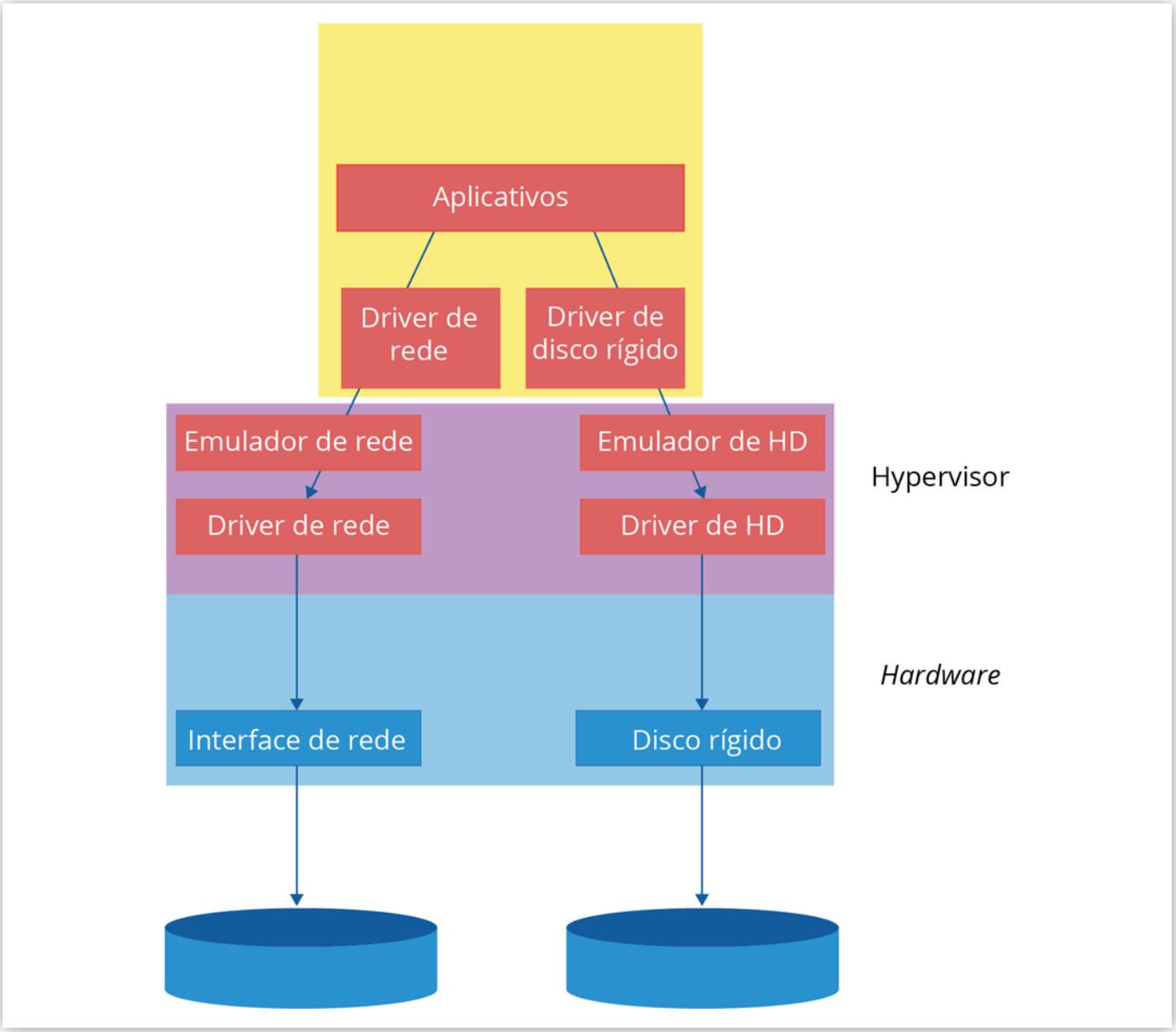


Figura 3.3 - Uma máquina virtual enviando requisições de acesso a dispositivos de armazenamento de rede e local
Fonte: Adaptada de Portnoy (2012, p. 153).

Nota-se, portanto, que essas operações não são executadas pela máquina virtual sobre o hardware em si, mas sim sobre as abstrações de hardware às quais têm acesso. Neste caso, este é o papel do hypervisor sobre a operação de armazenamento de dados.

Os discos reais, subjacentes aos discos virtuais podem ter características variadas, que não necessariamente se traduzem para o contexto virtualizado. Por exemplo, um servidor pode implementar uma tecnologia RAID para desenvolver formas de criar redundância de dados, ou um mesmo disco rígido de rede pode atender mais de uma máquina virtual que considera estar fazendo um acesso local.

saiba mais

Saiba mais

Discos rígidos podem ser organizados em uma arquitetura RAID para aumentar a confiabilidade de armazenamento por meio de redundância de informações entre discos, de forma a criar um sistema com tolerância a falhas. À exceção do chamado RAID 0 (em que apenas um disco rígido é utilizado, portanto, impossível realizar a redundância de equipamentos), o sistema foi desenvolvido para que, caso haja falha em algum dos discos rígidos do computador, as informações possam ser recuperadas de maneira automatizada.

Fonte: CBL TECH (2019).

Porém, é necessário sempre lembrar que os discos virtuais estão limitados

pelo sistema hospedeiro. Quando dito desta forma pode parecer natural que seja assim, afinal, um sistema com 100 GB de espaço de armazenamento não poderá armazenar mais que 100 GB de informações.

Para a criação de discos virtuais, entretanto, é necessário manter em mente quantas máquinas virtuais estarão hospedadas no mesmo sistema físico e como será realizada a alocação de disco para estas máquinas virtuais.

Para criar um disco rígido virtual, existem duas grandes estratégias, a estratégia de *Thin Provision* e *Thick Provision* , que que se subdivide em *Thick Provision Eager Zeroed* e *Thick Provision Lazy Zeroed* . Existe, ainda, uma técnica denominada *Raw Device Mapping* , em que a máquina virtual utiliza o próprio disco rígido hospedeiro como seu disco rígido virtual.

No caso de discos virtuais *Thick Provision* , os discos têm seus tamanhos especificados pelo administrador do sistema e todo o espaço necessário para o disco virtual é alocado no disco real.

Considere-se um exemplo de um conjunto de máquinas virtuais executadas em um servidor. Caso o administrador deseje, é possível dividir o disco rígido da máquina em secções estáticas que somem o espaço total disponível do disco. Por exemplo, um administrador de rede decide que é necessário instalar duas máquinas virtuais, cada uma com 200 GB de disco virtual.

Neste caso, são criados dois discos de 200 GB, o espaço em disco é alocado para estes dispositivos virtuais e o sistema hospedeiro terá 400 GB a menos de espaço de armazenamento para utilizar para outros fins.

Ademais, na estratégia de *Thick Provision Eager Zeroed* , não somente o espaço do disco virtual é alocado, mas tem o valor zero escrito em toda sua extensão. Na estratégia *Thick Provision Lazy Zeroed* , entretanto, só é escrito o valor zero no disco real quando um dado for inserido no disco virtual.

No caso da estratégia de *Thin Provision* , o administrador do sistema deve especificar a capacidade máxima dos discos a ser utilizados, mas, na verdade, os discos utilizam somente o espaço que for necessário. Neste caso os discos virtuais ocupam mais ou menos espaço no armazenamento do sistema

hospedeiro, dependendo da utilização dada à máquina virtual, desde que não seja ultrapassado o valor máximo definido pelo administrador do sistema.

Portanto, é possível criar máquina virtuais que possuam no máximo 200 GB de espaço de armazenamento, mas que não devam utilizar mais espaço de armazenamento do hospedeiro que o estritamente necessário. Neste caso, caso haja falha no planejamento do sistema de virtualização, pode ser que as máquinas virtuais tentem alocar mais armazenamento que o possível. Considerando-se o mesmo exemplo explicado anteriormente, suponha-se um hospedeiro que tenha 500 GB de memória e o administrador já tenha criado duas máquinas virtuais utilizando a estratégia de *thin provision* com 200 GB de armazenamento. Porém, cada máquina virtual somente alocou 50 GB de armazenamento e o administrador decide criar mais uma máquina com 200 GB de armazenamento que também alocou somente 50 GB. Esta situação é ilustrada na situação 1 da Figura 3.4.

Nesta situação, se todas as máquinas virtuais ocuparem o espaço total de armazenamento, o conjunto tentará utilizar 600 GB - mais do que o disponível no sistema. E o administrador não perceberá o erro até ser tarde demais. Esta situação é ilustrada na situação 2 da Figura 3.5.

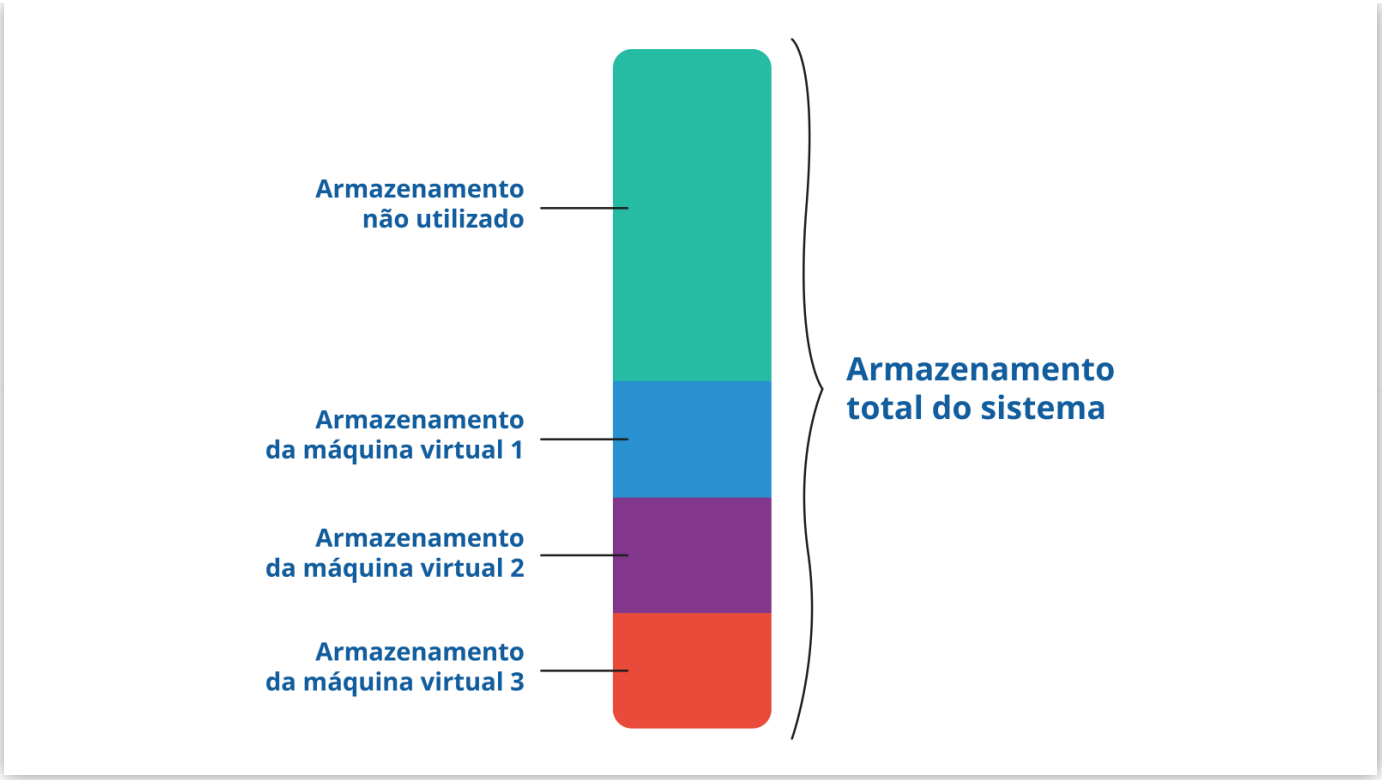


Figura - 3.4 - Situação 1

Fonte: Elaborada pelo autor.

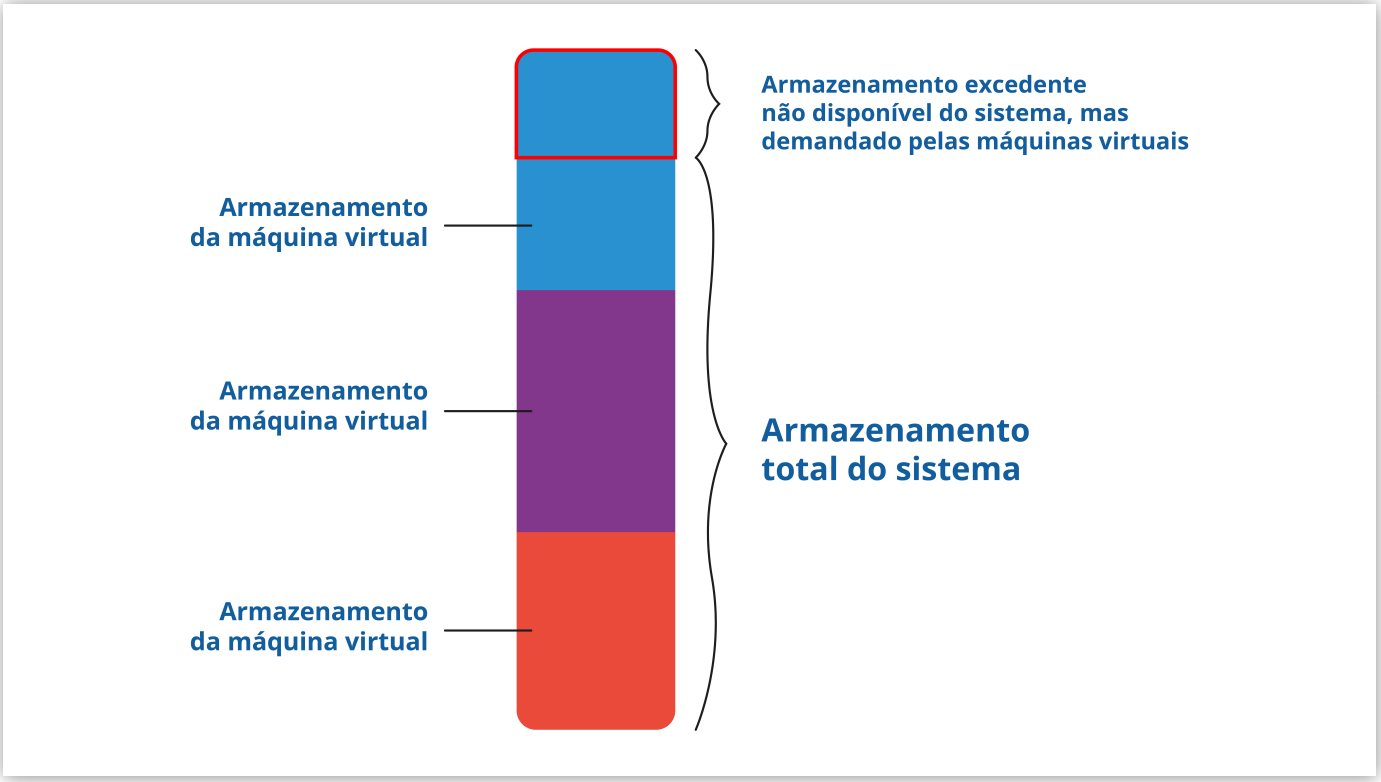


Figura - 3.5 - Situação 2

Fonte: Elaborada pelo autor.

Uma terceira forma de se criar discos rígidos virtuais é denominada *Raw Device Mapping* , em que se cria um disco rígido que age como proxy do disco

rígido físico. Quando a máquina virtual solicita a abertura de um arquivo, este proxy traduz o endereço de memória do disco rígido virtual para um endereço do disco hospedeiro. Para as operações de leitura e gravação, a máquina virtual se refere diretamente aos arquivos no disco rígido físico. Um diagrama deste processo se encontra na figura 3.6.

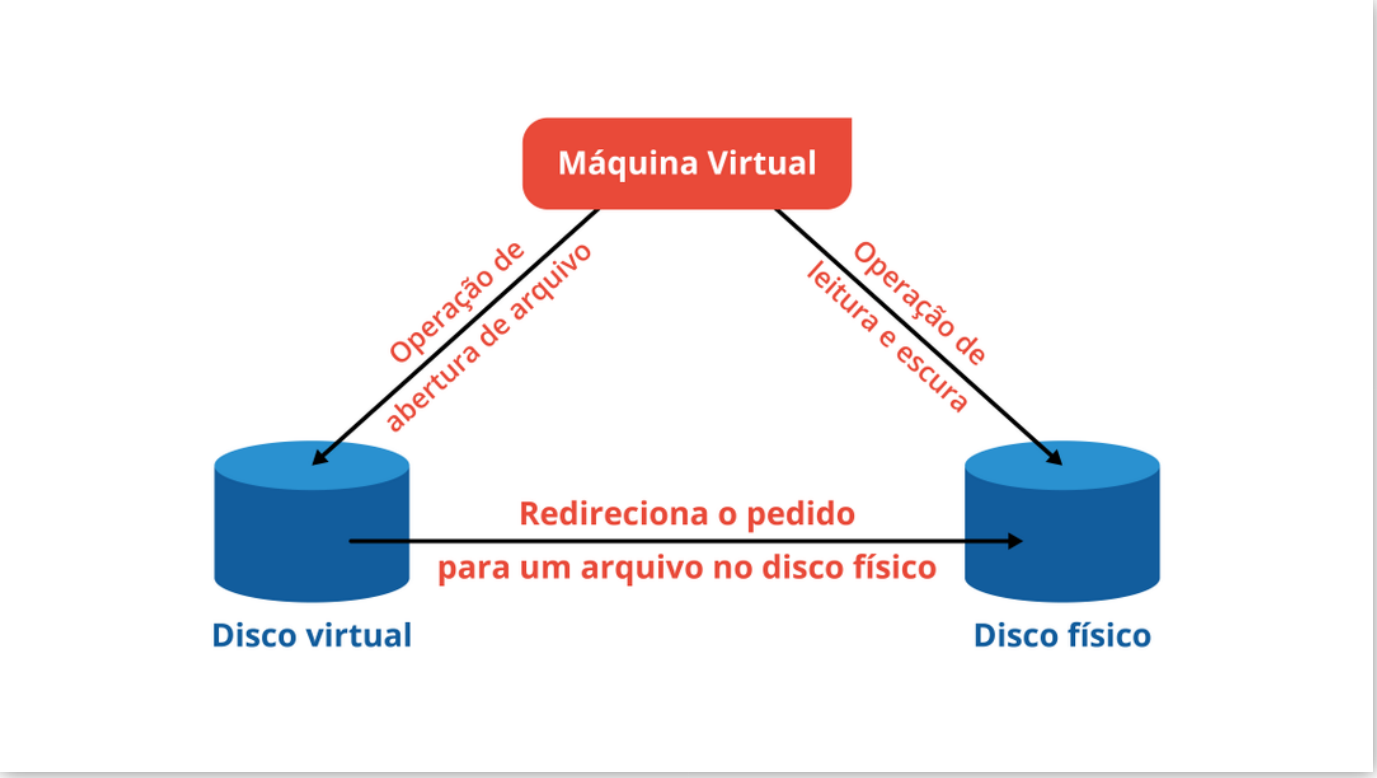


Figura 3.6 - Processo de leitura a escrita de arquivos utilizando Raw Device Mapping

Fonte: Adaptado de Vmware Docs (2019).

As diferenças entre as diferentes estratégias de alocação de armazenamento em virtualização estão no Quadro 3.1, a seguir.

Estratégia	Descrição
Thick Provision Eager Zeroed	O espaço em disco é alocado no momento da criação do disco virtual e zera em toda a extensão do disco virtual.
Thick Provision Lazy Zeroed	O espaço em disco é alocado no momento da criação do disco virtual e somente a porção do disco que será utilizada é zerada no momento da utilização.
Thin Provision	Define-se o espaço máximo que o disco virtual vai ocupar no disco físico. O tamanho do disco virtual aumenta ou diminui conforme o estritamente necessário para armazenar os dados.
Raw Device Mapping	O disco virtual mapeia o disco físico.

Quadro 3.1 - Comparação entre os tipos de alocação de espaço de armazenamento em virtualização.
Fonte: Adaptado de Vmware Docs (2019).

Outro problema com o qual se deve ter cuidado advém do fato de que todas as máquinas virtuais compartilham o mesmo armazenamento físico. Ou seja, apesar de todas as máquinas virtuais considerarem que têm um disco rígido inteiro para si, estão na verdade compartilhando o mesmo dispositivo físico e dividindo a velocidade de acesso entre si.

reflita

Reflita

Neste tópico foram discutidas as formas de criação de discos virtuais e a forma com que são organizados em ambientes virtualizados. Assim como com o processador, este componente virtual tem forte ligação ao dispositivo físico da máquina. A qualidade da virtualização, neste contexto, é dependente das tecnologias implementadas em hardware. Assim como a máquina virtual pode sofrer caso não exista redundância de dados por meio da tecnologia RAID, a máquina virtual não poderá ver múltiplos drives como um só. Até quanto a máquina virtual é independente de *hardware* ? Até que ponto o *hardware* pode compensar as falhas da máquina virtual?

Fonte: Elaborado pelo autor.

Portanto, se um dos sistemas hóspedes estiver realizando mais acessos ao disco que os demais, pode ser que acabe monopolizando todo o acesso a disco do sistema, aplicando uma redução de velocidade a todos os outros sistemas hóspedes ou até mesmo impedindo o acesso ao recurso por outras máquinas virtuais.

praticar

Vamos Praticar

Armazenamento é um recurso em constante expansão. [...] Uma razão para isso é que o tipo de dados com que se trabalha atualmente é diferente ao passado. Originalmente, somente informação de texto, palavras e números, eram armazenadas e processadas. Atualmente, encontram-se: clipes de vídeo e animações; gráficos, estáticos e animados; música e fala; além da informação textual apresentada em uma das cores e tamanhos disponíveis para web designers atualmente (PORTNOY, 2012, p.151, tradução nossa).

Com relação a sistemas de armazenamento virtualizados, é correto afirmar que:

- **a)** Máquinas virtuais não armazenam dados em disco. Quando são carregadas, utilizam espaço da memória do computador como armazenamento.
- **b)** O hypervisor verifica se o tamanho máximo dos discos rígidos virtuais é compatível com o espaço instalado no sistema hospedeiro para evitar erros futuros.
- **c)** Não importa quantos discos o sistema hospedeiro tenha, a máquina virtual ficará sempre isolada para um deles, sem se comunicar com os demais.
- **d)** Discos virtualizados são somente de leitura, uma vez que máquinas virtuais são estáticas e devem ser somente acessadas para consultas.
- **e)** Os discos virtualizados podem ter tamanho predefinido ou podem expandir até um tamanho máximo durante a utilização, dependendo da configuração da máquina virtual.

Memória Virtual em Máquinas Virtuais

Segundo Portnoy (2012), um dos recursos mais importantes de um sistema computacional é a memória, pois ela é um dos principais componentes que determina quão bem uma máquina virtual vai executar.

Para entender a influência que a memória tem no desempenho da máquina virtual, é necessário primeiro entender o conceito de hierarquia de memória. Um conceito que relaciona de maneira hierárquica as memórias presentes nos sistemas computacionais, sendo: 1. Memória cache; 2. Memória RAM; e 3. Disco rígido, que foi o assunto do tópico 2 desta unidade.

Na sequência anteriormente demonstrada, as memórias estão organizadas por velocidade de acesso: a memória *Cache* é a mais rápida, mas é a mais limitada em questão de quantidade disponível, normalmente na casa dos megabytes. A memória RAM tem velocidade intermediária e tamanho também intermediário, normalmente na casa das dezenas de gigabytes. Finalmente, o disco rígido é a memória com maior tempo de acesso, mas é também a mais abundante, indo de centenas de gigabytes até um ou dois terabytes.

A memória *cache* localiza-se dentro da CPU do computador, e ela possui o

menor tempo de acesso. Neste espaço de memória armazenam-se os dados que o microprocessador necessita para a execução imediata das atividades necessárias. Este espaço de memória já possui os próximos passos que a CPU deve executar e todos os valores de operandos relevantes.

Logo a seguir, na hierarquia de memórias, existe a memória RAM, com um tempo de acesso intermediário e com espaço também intermediário. Este elemento funciona na prática como um buffer para a memória *Cache* , todos os dados que eventualmente vão para o microprocessador passam antes pela memória RAM.

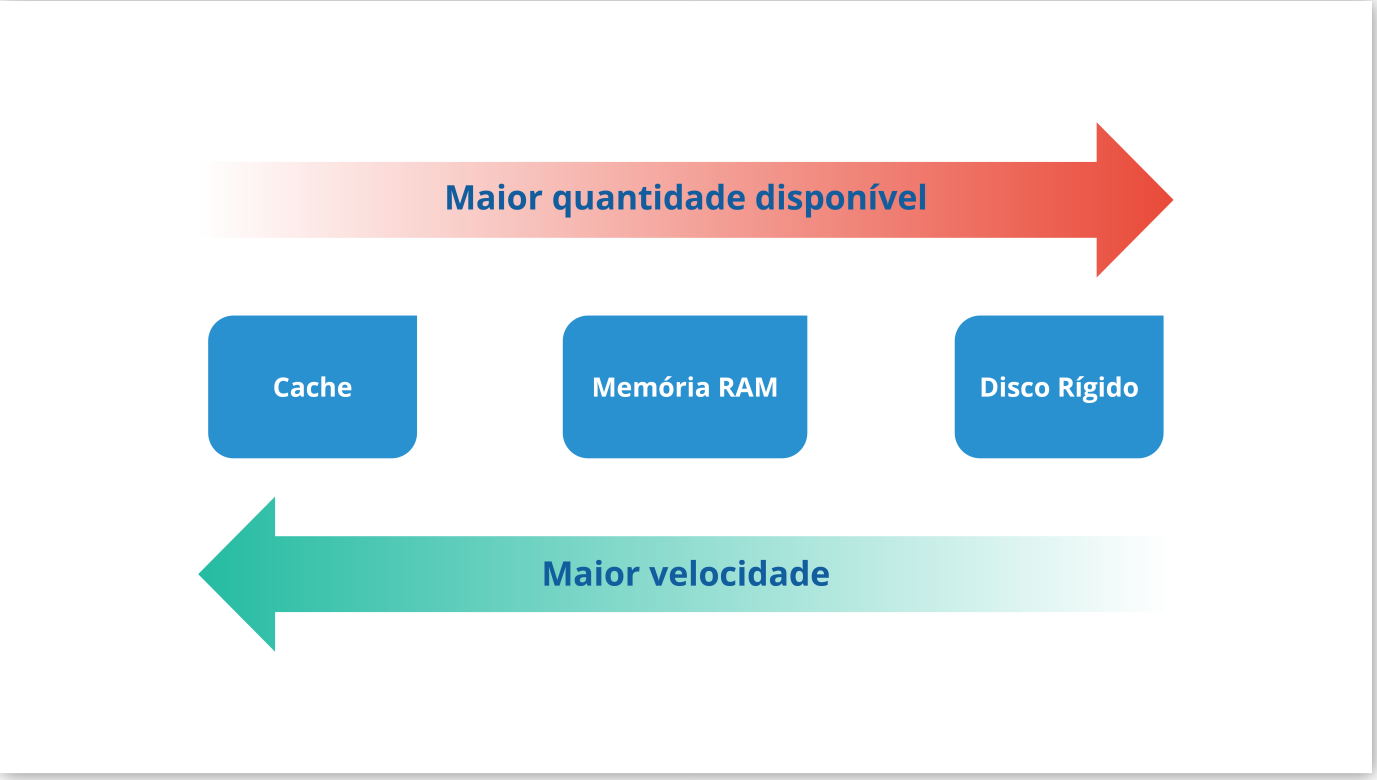


Figura 3.7 - Hierarquia de memória de um sistema computacional.
Fonte: Adaptado de Portnoy (2012, p. 153).

No contexto virtualizado, entretanto, devem ser administradas as memórias virtual e física do sistema de maneira similar ao armazenamento. Como aconteceu com o armazenamento, a memória virtual é apenas uma abstração da memória física do sistema.

Como com disco rígido, a quantidade de memória RAM virtual que estará disponível para uma máquina virtual é definida durante sua criação, mas pode ser alterada posteriormente. E, assim como o disco rígido também, o espaço

alocado pode ser tanto estático, definido durante a criação da máquina virtual, quanto ter somente o limite superior definido e a máquina virtual poderá utilizar até uma determinada quantidade.

Porém, como o uso de memória é vital, alguns métodos de otimização deste recurso foram implementados diretamente nos hypervisores, a fim de tanto minimizar os acessos à memória quanto a quantidade de memória utilizada. Mas a ideia fundamental de todos eles é que já que o hypervisor tem acesso a todos os dados que entram e saem das máquinas virtuais, pode interferir nessas transmissões para realizá-las de maneira mais eficiente.

Ao contrário do que acontece com o armazenamento virtual, a utilização de memória RAM virtual é sempre o mínimo possível, podendo, desta forma, compartilhar a memória entre várias plataformas virtualizadas. Consideremos um sistema computacional com 8 GB de memória e duas máquinas que estejam configuradas para utilizar 8 GB cada, mas que estejam utilizando apenas 2 GB. Dessa forma, na prática, estariam sendo utilizados apenas 4 GB, mesmo que a quantidade máxima de memória para virtualização seja de 16 GB.

Esse, entretanto, não é o único recurso que o hypervisor tem para a otimização do uso de memória. Uma das formas que o hypervisor pode atuar sobre o uso de memória é por meio do controle de cópias de dados armazenados, por meio de *data deduplication* .

Com essa técnica, o hypervisor verifica se um determinado conteúdo já está na memória. Caso esteja, sempre que uma máquina virtual precisa deste dado o hypervisor fornece um ponteiro para este e não o repete na memória, evitando, assim, duplicidades.

Essa técnica, porém, pode ser potencializada quando mais de uma máquina virtual que esteja executando em um sistema computacional hospedeiro tenha especificações similares de *software* . Caso o mesmo sistema operacional, por exemplo, esteja executando em dois ou mais sistemas hóspedes, é possível ter na memória apenas uma cópia dos dados, evitando armazenar dados em duplicidade.

Caso seja necessário, o sistema pode tratar seu armazenamento como uma memória secundária. O sistema operacional realiza esta operação, utilizando um pouco do disco como uma extensão da memória - chamada memória swap.

Em virtualização, há um processo denominado *balooning* , ilustrado na Figura 3.8.

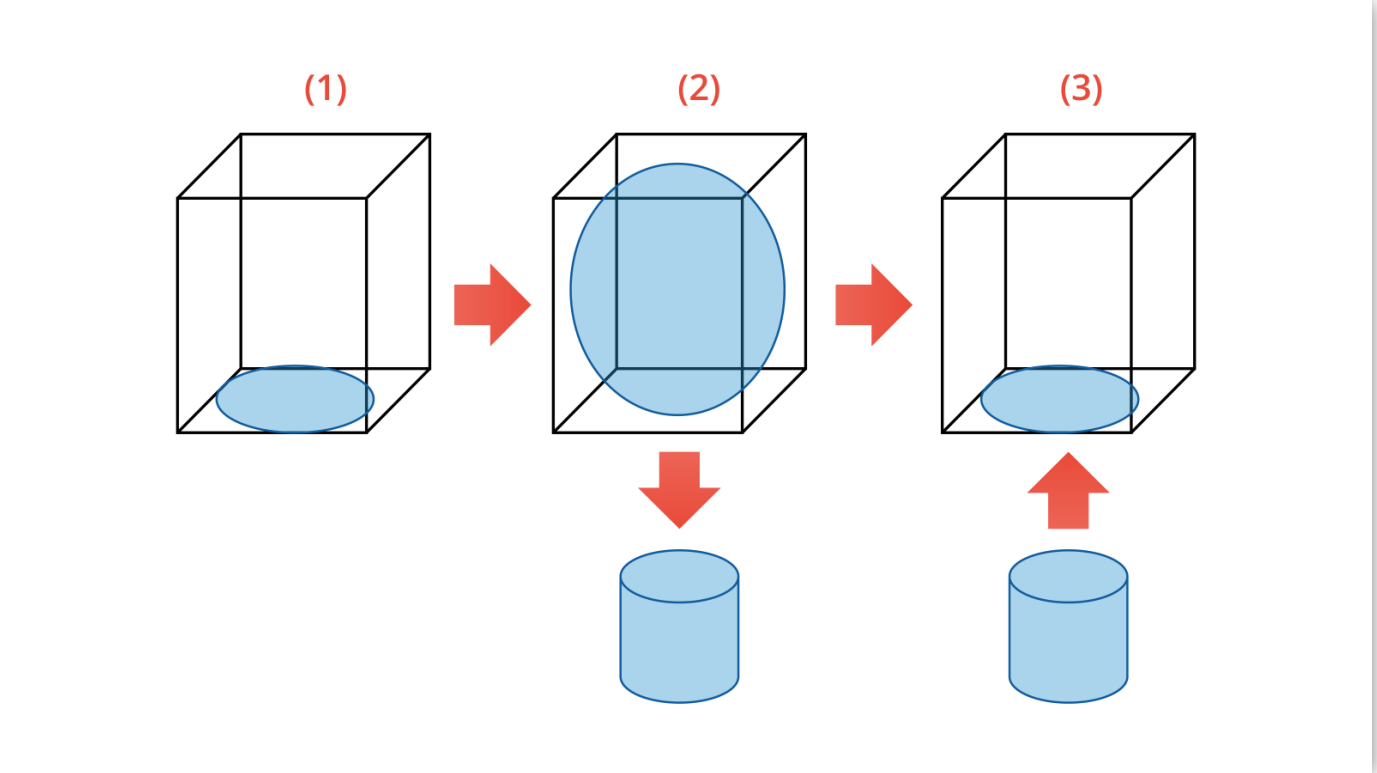


Figura 3.8 - Hierarquia de memória de um sistema computacional
Fonte: Adaptado de Portnoy (2012, p. 153).

O nome vem da analogia de um balão dentro de uma caixa, que deve ser esvaziado toda vez que atinge um determinado limite.

Na etapa (1) da Figura 3.8, a memória está se preenchendo e assim continuará até não haver mais espaço. Neste ponto, ilustrado pela etapa (2) da figura, o conteúdo é despejado em disco à espera da oportunidade para ser recolocado em memória. Finalmente, quando o sistema possui memória disponível suficiente, os dados são transferidos do disco para a memória novamente, conforme ilustrado na etapa (3).

praticar

Vamos Praticar

A memória é a área de trabalho do computador. Quando um sistema operacional inicializa, certas rotinas frequentemente utilizadas são carregadas na memória e ficam lá. Conforme os programas vão sendo executados, essas rotinas são copiadas para a memória [...]. Quando programas trabalham com informação, esses dados são movidos para a memória, a fim de que sejam transferidos rapidamente para a CPU e são escritos em memória novamente depois de qualquer transformação (PORTNOY, 2012, p.138, tradução nossa).

Com relação à memória da máquina virtualizada, é correto afirmar que:

- **a)** É possível que várias máquinas virtuais compartilhem os mesmos dados presentes na memória, caso o compartilhamento de paginação esteja ativado.
- **b)** A memória da máquina virtualizada não se comunica com o disco rígido sob nenhuma hipótese.
- **c)** A memória virtualizada não tem correspondência com a memória do sistema hospedeiro. A memória virtualizada é colocada em disco e acessada em disco quando necessário.
- **d)** O hypervisor não permite que a memória virtualizada total seja maior que a instalada no sistema hospedeiro e essa verificação é feita durante a criação de máquinas virtuais.
- **e)** Através do mecanismo de compartilhamento de páginas é possível realizar a comunicação entre máquinas virtuais.

Armazenamento em Virtualização

Uma das aplicações de virtualização é a criação de sistemas de armazenamento on-line para grandes volumes de dados, que dão suporte às aplicações de Big Data.

As tecnologias apresentadas nesta unidade são utilizadas para implementar sistemas de armazenamento em nuvem. Porém, apesar de serem necessárias não são suficientes, já que ainda são necessários os serviços de armazenamento e de banco de dados.

Para atender essas necessidades, algumas empresas desenvolveram plataformas que ao serem instaladas fornecem essas funcionalidades. Porém, serão tomadas para estudos a plataforma em nuvem BigTable da Google e a suite de softwares Pig, HBase e Hive da Apache.

Google BigTable

Desenvolvida pela Google, a plataforma BigTable é um servidor NoSQL fornecido pelo Google e que executa nos servidores da empresa, ou seja, externamente ao ambiente do cliente. Inclusive, a contratação do serviço já

tem como um dos campos de atuação do Google a manutenção dos serviços de infraestrutura.

O sistema é dividido em três componentes: 1. A biblioteca, que é ligada a um cliente; 2. Um servidor mestre; e 3. Vários servidores de *tablets* . Sendo este último o componente que garante o funcionamento do sistema como um todo.

A arquitetura geral do sistema se baseia na unidade de armazenamento denominada *tablet* , que pode ser transferido entre servidores de armazenamento conforme a conveniência para a aplicação. Inicialmente, cada *tablet* contém uma tabela do banco de dados de um cliente. Porém, conforme a quantidade de dados cresce, uma tabela pode ser armazenada em mais de um *tablet* .

O ponto de entrada é um arquivo do tipo *Chubby* , uma implementação de um sistema de arquivos distribuídos para aplicações de *Cloud Computing* . Cada cliente tem um arquivo deste tipo que tem a localização do *tablet* raiz, que por sua vez aponta para *tablets* de metadados, que apontam para os tablets que contêm as tabelas do usuário em questão. (CHANG *et al* ., 2008)

Para o usuário do sistema, entretanto, este funcionamento é transparente. O usuário se preocupa somente com o modelo de dados, composto de *timestamps* , linhas e famílias de colunas.

As linhas do modelo de dados armazenam as informações que o usuário pode ler ou escrever no sistema. Obrigatoriamente, cada linha tem os dados que são armazenados no banco de dados e uma chave que permite que sejam realizadas consultas.

Essas linhas são armazenadas nos tablets a fim de ser possível balancear a carga entre os servidores que executam a aplicação. Dessa forma, as informações sempre estão acessíveis para o usuário, mas ele não precisa ter informações sobre onde estas estão armazenadas.

Os nomes das colunas são armazenados em famílias de colunas, que contêm os nomes de uma ou mais colunas de um banco de dados. Organizando, desta

forma, os tipos de colunas conforme as famílias.

Finalmente existem os *timestamps* , que são inteiros de 64 bits que indicam o ponto no tempo em que os dados foram inseridos no banco de dados. Nesse tipo de aplicação é comum ter dados replicados, mas que foram incluídos em tempos diferentes. Para entender a utilidade do *timestamp* , considere um sistema em que usuários podem enviar arquivos para um servidor, mas somente um arquivo por vez é armazenado por cliente. Um mesmo usuário enviará diversos arquivos em tempos diferentes, ou seja, dados diferentes em *timestamps* diferentes, porém serão armazenados na mesma célula do banco de dados. Dessa maneira, a aplicação pode prover controle de versão para o usuário, por exemplo.

É importante ter em mente que este serviço provê para o usuário apenas o armazenamento das informações. Para fazer análise desses dados, existem outras ferramentas, desenvolvidas pelo Google ou outras empresas.

Apache

A fundação Apache criou um conjunto de *softwares* focados em *Big Data* e *Data Mining* , que implementam funcionalidades que permitem a administradores manter dados, analisar padrões e inferir conclusões a partir das operações anteriores.

Apache Hive

A plataforma Hive, da fundação Apache implementa uma forma de armazenar, recuperar e modificar grandes conjuntos de dados. O Hive executa sobre a plataforma Hadoop, também da Fundação Apache, que permite a criação de bases de dados SQL em ambientes com plataformas múltiplas.

Nessa plataforma, o modelo de dados é dividido em tabelas, partições e *buckets* . O primeiro, tabelas, é análogo às tabelas de banco de dados relacionais, a única diferença é que existe o suporte a tabelas externas, em que uma tabela pode ser criada a partir de arquivos preexistentes.

Partições correspondem a como os dados são armazenados no banco de dados. Esta definição é importante para possibilitar a busca de tipos de dados nesta base de dados. Por exemplo, se fosse criada uma partição para dados do tipo data, seria possível buscar somente datas em uma base de dados.

Finalmente *buckets* são tipos de dados que armazenam divisões das partições e podem ser criados a partir de buscas na base de dados, utilizando uma operação *SAMPLE* , em que apenas um fragmento dos dados é recuperado da base.

Apache HBase

Segundo a Apache (2019), a plataforma HBase, da Fundação Apache, assim como a plataforma Hive, permitem a criação de bancos de dados com grande número de linhas e tabelas. A principal diferença entre as duas plataformas é que enquanto o Hive é um banco de dados distribuído SQL, o HBase é um banco de dados distribuído NoSQL.

Esta plataforma é fortemente baseada no sistema BigTable desenvolvido pelo Google, com a arquitetura de dados dividindo as informações em *shards* (um conceito similar ao *tablet* da plataforma do Google) e movendo esses *shards* entre os servidores da aplicação de maneira transparente ao usuário.

A grande diferença entre as plataformas, entretanto, reside no fato de que o Apache HBase é distribuído para ser instalado nas máquinas do cliente que deseja o serviço (e que tem que se responsabilizar pela integridade dos dados e serviços também). Neste ambiente, o cliente deve criar sua própria infraestrutura de servidores e realizar a instalação dos softwares, enquanto com o Google, o usuário somente contrata o serviço de computação em nuvem.

Apache Pig

Para realizar a análise de dados sobre volumes grandes de informação, a Fundação Apache criou o Pig, que é uma plataforma que permite a busca e o processamento de informações em ambiente de *Big Data* .

Esse ambiente provê ao usuário diferentes funcionalidades para fracionar,

realizar buscas e filtrar dados em ambientes distribuídos.

Todas as plataformas apresentadas neste tópico se beneficiam da virtualização de maneira direta ou indireta. As plataformas da Fundação Apache podem ser instaladas em ambientes virtualizados de forma a abstrair o ambiente distribuído, tratando um conjunto de plataformas de hardware como se fosse uma só.

No caso do BigTable, o provedor do serviço cria um ambiente virtualizado para cada usuário, que passa a perceber somente a secção do sistema que lhe diz respeito.

praticar

Vamos Praticar

Nos últimos dois anos e meio, foi desenvolvido no Google um sistema de armazenamento distribuído para lidar com dados estruturados chamado BigTable. O BigTable foi projetado [...] De várias maneiras, BigTable se parece com um banco de dados: a implementação usa muitas estratégias como as utilizadas em banco de dados. Bancos de dados paralelos e bancos de dados *main-memory* alcançaram escalabilidade e alta performance (CHANG *et al* ., 2008, p.1, tradução nossa).

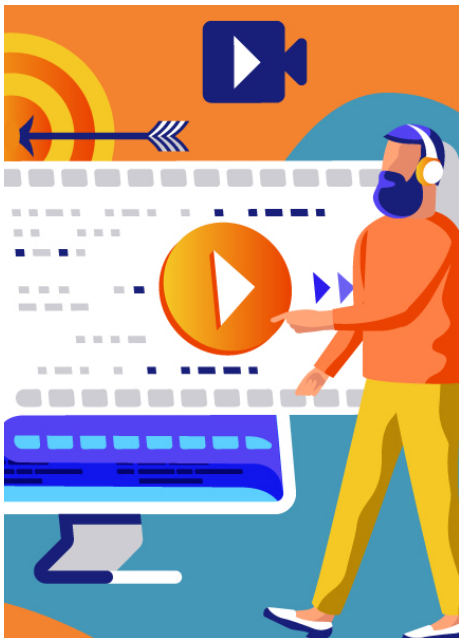
Assinale a alternativa correta:

- **a)** O sistema BigTable está disponível para ser instalado em máquinas proprietárias, não dependendo do desenvolvedor da plataforma.
- **b)** As plataformas da Fundação Apache não podem ser instaladas em máquinas proprietárias, devendo ser utilizadas como um serviço de nuvem.

- **c)** As plataformas de armazenamento de dados, HBaase, Hadoop e BigTable não possuem uma forma padronizada de buscar informações.
- **d)** A plataforma Pig armazena dados em formato NoSQL.
- **e)** Apesar de ser uma linguagem antiga, o SQL ainda é usado em todas as plataformas.

indicações

Material Complementar



FILME

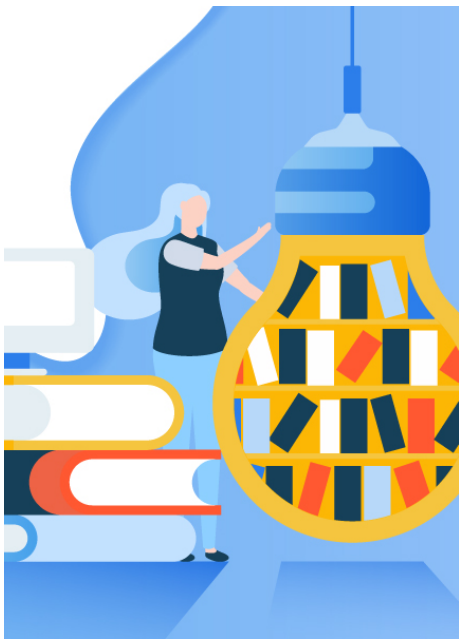
Nome : Jobs

Ano : 2013

Comentário : Um filme que retrata o início da Apple e como Steve Jobs e Steve Wosniak fundaram a empresa de tecnologia e, posteriormente, como a Apple se desenvolveu. Um filme sobre como a visão de negócios se aplica à informática e como produtos e serviços centrados no usuário podem ser criados.

Para conhecer mais sobre o filme, acesse o trailer disponível em:

TRAILER



LIVRO

Nome do livro : Computação em Nuvem

Editora : Brasport

Autor : Manoel Veras de Souza Neto

ISBN : 8574527475

Comentário: Este livro aborda questões de infraestrutura e de arquitetura para computação em nuvem. Para ilustrar o desenvolvimento das questões, o autor desenvolve um comparativo entre as arquiteturas empresariais e de computação em nuvem, abordando de maneira didática as divisões e responsabilidades das infraestruturas.

conclusão

Conclusão

Nesta unidade, foram feitas considerações sobre como o *hardware* virtual funciona, os algoritmos utilizados para a otimização e simulação dos dispositivos virtuais e como devem ser as configurações da máquina virtual para que todos os componentes trabalhem de forma harmônica.

É importante que esses tópicos sejam levados em consideração tanto durante a etapa de projeto do sistema virtual quanto durante a operação e manutenção dos serviços virtuais, de forma a evitar erros que possam limitar o desempenho da máquina virtual e atender os clientes da forma mais eficiente possível.

Nota-se com relativa facilidade que as técnicas utilizadas no hardware virtual são similares às utilizadas em dispositivos físicos, durante a execução rotineira, e que foram transpostas para o contexto virtualizado, sofrendo as adaptações cabíveis para a utilização neste novo ambiente.

Finalmente, foi possível verificar que, apesar de o hardware virtual abstrair o físico e gozar de uma certa independência, o vínculo entre os dois ainda assim é estreito e deve ser levado em consideração para que o sistema virtual opere de maneira satisfatória.

referências

Referências

Bibliográficas

APACHE. **Apache HBase Reference Guide** , s.d. Disponível em: https://hbase.apache.org/book.html#arch.overview . Acesso em: 25 jan. 2020.

APACHE. Apache HIVE. **Design** , 2015. Disponível em: https://cwiki.apache.org/confluence/display/Hive/Design . Acesso em: 25 jan. 2020.

APACHE. Apache Hadoop. **Getting Started** , 2017. Disponível em: http://pig.apache.org/docs/r0.17.0/start.html . Acesso em: 25 jan. 2020.

CAMARGO, R. F. **O que é Overhead?** Conheça a importância de controlar os custos indiretos. Treasey. Santa Catarina, 2017. Disponível em: https://www.treasy.com.br/blog/overhead/ . Acesso em: 25 jan. 2020.

CBL TECH. **Blog CBL Recuperação de Dados** . O que é RAID? Quais são os tipos de RAID?, [s.l.], 2019. Disponível em: https://cbltech.com.br/blog/o-que-e-raid-tipos.html . Acesso em: 25 jan. 2020.

CHANG, F. *et al* . Bigtable: A distributed storage system for structured data. **ACM Transactions on Computer Systems (TOCS)** . Nova York, v.26, n.2, p.4-18, jun. 2008.

JOBS . Jobs. Direção Joshua Michael Stern. Elenco: Ashton Kutcher, Dermot Mulroney e Josh Gad. EUA, 2013. Filme. Disponível em: https://www.youtube.com/watch?v=pT2Rp7c9Qok&feature=youtu.be . Acesso em: 21 jan. 2020

PORTNOY, M. **Virtualization Essentials** , 1. ed. Indianapolis, Indiana: John Wiley & Sons, Inc., 2012.

UNIONPÉDIA COMUNICAÇÃO. Google Play. **Overhead (computação)** , 2019. Disponível em: https://pt.unionpedia.org/i/Overhead_(computa%C3%A7%C3%A3o) Acesso em: 25 jan. 2020.

VERAS, M. **Computação em nuvem** . Rio de Janeiro: Brasport, 2015.

VMWARE DOCS. **About Virtual Disk Provisioning Policies** . VMare vSphere 5.1 Documentation Center. s.d. Disponível em: https://pubs.vmware.com/vsphere-51/index.jsp?topic=%2Fcom.vmware.vsphere.storage.doc%2FGUID-4C0F4D73-82F2-4B81-8AA7-1DD752A8A5AC.htm . Acesso em: 25 jan. 2020.

VMWARE DOCS. **About Raw Device Mapping** . VMware vSphere 6.0, 2019. Disponível em: https://docs.vmware.com/en/VMware-vSphere/6.0/com.vmware.vsphere.storage.doc/GUID-9E206B41-4B2D-48F0-85A3-B8715D78E846.html . Acesso em: 22 jan. 2020.