Sistemas Virtualizados – Uma visão geral

Luciane Politi Lotti
RA 012096
lotti@unicamp.br

Alysson Bolognesi Prado
RA 931544
alysson@unicamp.br

RESUMO
O presente trabalho possui como objetivo descrever o contexto evolutivo dos sistemas virtualizados no que tange aos seus conceitos, diferenças de implementação, aplicações, utilizações, tendências e desafios. Foi dada ênfase especialmente na importância do seu uso como condição basilar para o imenso campo vislumbrado para as novas tecnologias de informação e comunicação que centralizam suas possibilidades de acessos nos indivíduos e não mais nas máquinas. Paradoxalmente, segurança, escalabilidade, eficiência, economia e sustentabilidade passam a constituir os grandes desafios a serem enfrentados pela virtualização.

Categorias e Descritores de Conteúdo
C.1.0 [Processor architectures]: General; D.4.1 [Operating systems] Process Management- Multiprocessing/ multiprogramming/ multitasking.

Termos Gerais
Management, Design, Theory.

Palavras-chave
Virtualização, Sistemas Virtualizados, Hypervisor, Interface Hardware-Software.

1. INTRODUÇÃO
Virtualização é um termo amplo que se refere à abstração de recursos computacionais, ocultando dos seus usuários algumas de suas características sejam eles pessoas, programas ou sistemas operacionais.

Através do uso de virtualização, um único computador físico, ou *host*, pode compartilhar seus recursos de hardware entre múltiplos sistemas operacionais hospedados, ou *guests*, cada um percebendo a si mesmo como tendo controle total sobre sua máquina virtual [9] ou ainda, em alguns casos, como se estivesse instalado isoladamente em uma máquina física.

A abordagem mais utilizada é através da adoção de um software que atua como mediador entre as instâncias de diversos sistemas operacionais executados simultaneamente em uma mesma máquina física [7]. Este software é chamado Virtual Machine Monitor (VMM) ou *hypervisor* e é responsável por garantir o isolamento entre as máquinas virtuais, a limitação dos recursos destinados a cada uma e o acesso aos dispositivos (Figura 1).

MO401-1S2011, Instituto de Computação - Unicamp.

Os primeiros sistemas baseados em virtualização surgiram em meados da década de 1960 [2]. Por exemplo, o sistema CP-40 provia máquinas virtuais compatíveis com o System/360 da IBM. Durante a década de 1970 foram pesquisados e estabelecidos os princípios teóricos que norteiam, até hoje, o projeto de arquiteturas computacionais que suportam adequadamente a virtualização [10].

Atualmente, com a adoção pelos fabricantes da arquitetura x86 de primitivas necessárias à virtualização, esta deixou de ser restrita a computadores de grande porte e se popularizou, atingindo tanto computadores *desktop* quanto servidores e *clusters* de processamento.

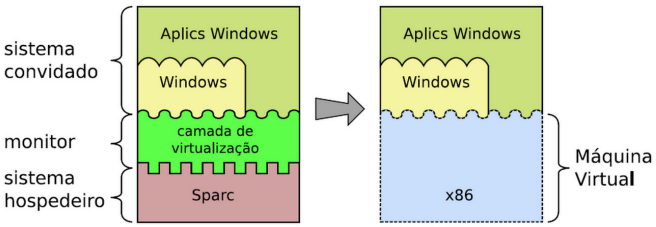

Figura 1: um sistema virtualizado se comporta como um sistema real, não há diferença estrutural perceptível para as aplicações em execução [18]

Neste artigo, revisamos na Seção 2 os principais conceitos envolvidos com a virtualização. Na seção 3 resgatamos sua definição, requisitos e implementação na arquitetura x86. Na Seção 4 apresentamos diversas formas de virtualização atualmente em uso, seguindo de uma coletânea de motivos para sua adoção na Seção 5, finalizando na Seção 6 com tendências de pesquisa e desafios futuros para sistemas virtualizados.

2. ABSTRAÇÃO E VIRTUALIZAÇÃO
Primeiramente, devemos distinguir duas possibilidades de virtualização, entre máquinas virtuais e sistemas virtualizados: as máquinas virtuais de aplicação (*Process Virtual Machines*) são destinadas a suportar apenas um processo ou aplicação convidada específica; as máquinas virtuais de sistemas (*System Virtual Machines*) são destinadas a suportar sistemas operacionais convidados completos e suas aplicações [19].

A segunda diz respeito à diferença entre abstração e virtualização. Na abstração é fornecida uma interface de acesso homogênea e simplificada aos recursos do sistema cuja execução de aplicativos se dá diretamente em "arquivos"; na virtualização são criadas novas interfaces a partir das interfaces existentes com execuções em discos virtuais (Figura 1).

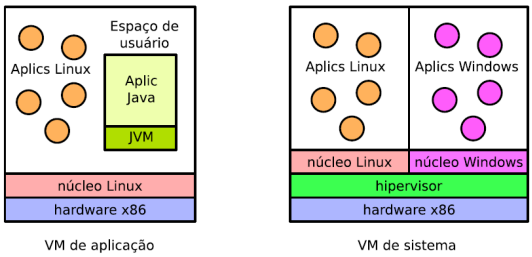

Figura 2 - VM's de aplicação e de sistema [18]

A primeira diferença implica em diferentes tipos de máquinas, de virtualização e de uso: emulação, virtualização, simulação, paravirtualização.

É justamente na diferença de utilização entre disco real e virtual que residem as principais propriedades e requisitos do *hypervisor* que, ao mesmo tempo, também constituem seus principais pontos de vantagens e desvantagens. Seriam: prover ambiente de execução virtual idêntico ao real (equivalência), leve degradação de desempenho (eficiência), controle de recursos, isolamento, gerenciabilidade, inspeção, encapsulamento e recursividade.

Tabela 1 - Classificação de Máquinas Virtuais de Sistema

	Virtualização de recursos	Virtualização completa
Monitores nativos	Xen,VMWare ESX, Parallels	Crusoe/Efficeon, IBM A/S400
Monitores convidados	UML, VirtualBox, kQEmu, VMWare Wstation, KVM	QEmu, VirtualPc, Bochs, Hercules

2.1. Simulação e Emulação

Uma forma de se obter um ambiente com comportamento semelhante ao de uma determinada máquina física e com controle sobre os recursos por ela utilizados é através de emulação. Em geral, usa-se para dar suporte a uma arquitetura legada por meio da tradução de cada instrução da máquina legada para a máquina em uso [13].

Simulação, por outro lado, se refere apenas a apresentar comportamento similar ao de um determinado ambiente, em relação a entradas e saídas, que pode ser útil para design, prototipação e testes. Estas soluções provocam uma grande perda de performance e, apesar de muito úteis em situações específicas, não apresentam a eficiência de um mecanismo verdadeiro de virtualização.

2.2. Virtualização Parcial

Na virtualização parcial ocorre a virtualização por espaços de endereçamento em que a máquina virtual simula várias instâncias de um grande ambiente de hardware subjacente. Normalmente, isto significa que os sistemas operacionais não podem ser executados inteiramente na máquina virtual (que seria o sinal de virtualização completa), mas sim apenas uma quantidade considerável de aplicativos pode ser executada. A principal forma de virtualização parcial é a virtualização do espaço de endereço, em que cada máquina virtual consiste de um estado independente. Esta funcionalidade exige transferência de endereço de hardware, e esteve presente na maioria dos exemplos práticos de virtualização parcial. A virtualização parcial foi um marco histórico importante no caminho para a virtualização completa.

A experiência com a virtualização parcial, e suas limitações, conduziu à criação do primeiro sistema completo de virtualização (IBM PC-40, a primeira iteração da CP/CMS, que acabaria por se tornar a família IBM VM) a mesma abordagem foi utilizada por sistemas mais recentes como o Microsoft Windows e o Linux.

A virtualização parcial é significativamente mais fácil de implementar do que a completa, é capaz de suportar aplicações importantes e se provou com sucesso em usos de recursos do computador com partilha entre vários usuários.

No entanto, em comparação com a virtualização completa, a sua desvantagem é em situações que necessitem de compatibilidade com versões anteriores ou portabilidade. Pode ser difícil de prever com precisão quais recursos serão usados por uma determinada aplicação. Se determinados recursos de hardware não são simulados, qualquer software usado nesses recursos poderá falhar.

2.3. Paravirtualização

Trata-se de outra alternativa de virtualização, que facilita a implementação do software necessário [7]. O sistema operacional da maquina *guest* tem conhecimento de que está operando em um ambiente virtualizado. Seu código é modificado, removendo todas as instruções que poderiam fazer acesso direto aos recursos físicos e comprometer a integridade das máquinas virtuais, e as substituindo por chamadas explícitas ao *hypervisor*.

Neste caso a camada adicional de software para virtualização é reduzida e a emulação do comportamento das instruções críticas eliminado, apresentando um desempenho da máquina virtualizada muito próximo ao de uma máquina não-virtualizada. Exemplos incluem IBM LPARs , Win4Lin 9x , Sun Logical Domains , z/VM e Trango.

2.4. Virtualização Completa

É uma técnica utilizada para prover uma máquina virtual do ambiente com simulação completa do hardware subjacente. Esta virtualização requer que cada característica do hardware seja devidamente refletida na máquina virtual, incluindo todo o conjunto de instruções de entrada e saída de operações, interrupções, acesso à memória e todos os demais elementos utilizados por um software que fosse executado em uma máquina real. De tal feita qualquer software deve ser capaz de ser executado na máquina virtual e, em particular, seus sistemas operacionais. A prova evidente de virtualização é dada pela medida em que um sistema operacional com êxito de uso *standalone* também repita o mesmo desempenho dentro de uma máquina virtual.

A virtualização completa só é possível dada a combinação certa de elementos de hardware e software como foi o caso da série IBM System/370 após a adição do hardware de memória virtual em 1972. A virtualização completa da plataforma x86 se deu em 2005-2006 com a adição de extensões da plataforma virtual do AMD-V e Intel VT-x. Outros exemplos incluem ADEOS, Mac-on-Linux, o Parallels Desktop para Mac, o Parallels Workstation, VMware Workstation, VMware Server (anteriormente GSX Server), o VirtualBox, Win4BSD e Win4Lin Pro.

Um desafio fundamental para a virtualização completa é a interceptação e a simulação de operações privilegiadas, tais como instruções de I/O. Os efeitos de cada operação realizada dentro de uma determinada máquina virtual devem ser mantidos nela mesma, já que suas operações não podem alterar o estado de qualquer outra máquina virtual, programas de controle ou hardware.

Existem basicamente duas formas de se realizar este particionamento. A primeira é através da instalação de um hypervisor diretamente na máquina e a partir dele criar máquinas virtuais e instalar sistemas operacionais independentes em cada uma delas (Figura 3). A segunda opção é a instalação de um sistema operacional hospedeiro diretamente no hardware, que comporta o VMM como um aplicativo específico que gerenciará as máquinas virtuais juntamente com demais softwares em execução no sistema *host* (Figura 4). Há autores [11] que fazem a distinção de nomenclatura, referenciando-se ao hypervisor especificamente para o primeiro caso e VMM para o segundo; em outras referências há uso indiscriminado de ambos os nomes.

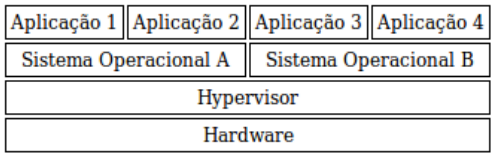

Figura 3: camada de virtualização diretamente no hardware [11]

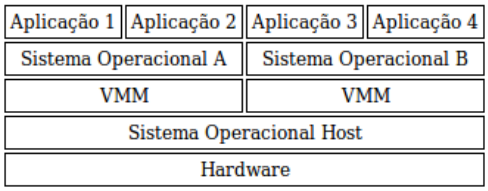

Figura 4: camada de virtualização instalada em um sistema operacional hospedeiro [11]

A instalação diretamente no hardware (*bare metal*), por um lado, garante um aumento no desempenho por haver menos intermediários entre o software em execução na máquina virtual e os dispositivos físicos usados (processador, memória, I/O), porém exige o desenvolvimento do software *hypervisor* mais complexo, fazendo uso de técnicas de emulação e simulação para dar a máquina virtual um contexto similar ao de uma máquina real. Por outro lado, a presença de um sistema operacional já instalado no *host* fornece uma série de primitivas de acesso ao hardware e controle de periféricos que facilita o desenvolvimento de um VMM com menos linhas de código e possibilidades de *bugs*.

3. IMPLEMENTANDO UMA ARQUITETURA VIRTUALIZÁVEL

A presença da possibilidade de virtualização deve ser uma característica a ser definida nos estágios iniciais do projeto de uma nova arquitetura. Caso contrário um esforço muito maior é necessário para que se acrescente esta opção em uma arquitetura já consolidada e em pleno uso. Desta forma, é importante que se conheçam os requisitos teóricos fundamentais para sua implementação.

3.1. Requisitos para Virtualização

Uma máquina virtual, conforme a definição de Popek e Goldberg [10], é uma "duplicata eficiente e isolada da máquina real" [12] que apresenta características de desempenho muito similares aos da máquina original, mas com restrições de acesso aos seus recursos físicos, que são gerenciados por um monitor de máquina virtual. Para que um computador seja verdadeiramente virtualizável, é necessário que o hardware forneça algumas características.

A primeira delas é a presença de pelo menos dois modos de operação, que chamaremos de modo supervisor, em que o conjunto completo de instruções do processador está disponível para execução, e um modo usuário, em que algumas instruções não devem ser executadas. Os programas acessados pelos usuários em geral são executados neste último modo.

Precisa-se também do conceito de *trap*, que consiste em o processador detectar situações em que deve parar o fluxo normal de processamento, armazenar o status completo da CPU (ponteiro de instruções, limites da memória, modo de execução atual), mudar para o modo supervisor e executar uma rotina armazenada em uma posição pré-definida da memória. Ao fim desta execução, o processador deve retornar ao status previamente armazenado.

Em seguida, define-se um conjunto de instruções chamadas privilegiadas, que são aquelas que, quando executadas em modo usuário, geram um *trap*, enquanto que se executadas em modo supervisor não geram *trap*. Outro conjunto de instruções, chamadas sensíveis, corresponde às que acessam ou alteram o modo como o processador está executando ou a quantidade de recursos por ele utilizados (ex: memória).

Com base nestes requisitos, Popek e Goldberg enunciam o seguinte **teorema**: um monitor de máquina virtual pode ser construído para uma determinada arquitetura se o conjunto de instruções sensíveis para este computador for um subconjunto de suas instruções privilegiadas.

Ou seja, em um sistema virtualizado, qualquer instrução que tentar acessar as definições dos recursos do sistema, se estiver sendo executada em modo usuário, gerará um *trap* que evocará o VMM. Este fará uso de mecanismos de emulação ou simulação para produzir o comportamento desejado, garantindo-se o isolamento entre as máquinas virtuais e o respeito pelos limites definidos para cada uma delas.

Como se garante que todas as instruções não-privilegiadas não são potencialmente prejudiciais à virtualização, elas podem ser diretamente executadas pelo processador, garantindo-se o requisito de desempenho muito próximo ao de uma máquina real.

3.2. Memória virtual

Pelo exposto na seção anterior, a presença de um mecanismo de memória virtual não é um requisito necessário à implementação de uma arquitetura virtualizável. Entretanto esta característica comum nos processadores atualmente em uso pode ser ao mesmo tempo um auxiliar e uma preocupação no uso de virtualização.

Da mesma forma que em um sistema virtualizado visa-se o isolamento entre diversos sistemas operacionais em execução simultânea, a memória virtual permite que processos compartilhem o hardware sem interferir um com o outro [12].

Na virtualização total, o *hypervisor* reserva um espaço de memória para cada sistema convidado, entretanto, o mapeamento da memória virtual é feito de tal forma que cada sistema *guest* enxerga sua respectiva memória iniciando no endereço 0x00, até o limite configurado pelo *hypervisor* [19]. No caso de paravirtualização os sistemas convidados, cientes da virtualização,

recebem do *hypervisor* a informação sobre quais áreas da memória podem utilizar.

O desafio está em arquiteturas que não têm identificação do processo em execução associado ao conteúdo do TLB [10]. Como cada sistema *guest* tem seu próprio gerenciamento da tabela de páginas da memória virtual, cada mudança no espaço de endereço normalmente exige um esvaziamento do TLB, implicando em perda de desempenho.

3.3. Virtualização de x86

Apesar de serem de uma das arquiteturas mais populares atualmente, os processadores da linha x86 não foram projetados, em sua origem, para serem virtualizados [7]. Apenas algumas das instruções sensíveis geravam *traps* quando executadas em modo usuário.

Por exemplo, a instrução PUSHFD coloca na pilha um conjunto de *flags* do processador que podem permitir a um sistema operacional detectar que está sendo executado em uma máquina virtual. Esta instrução não é interceptada quando executada em modo usuário. As primeiras tentativas de implementar máquinas virtuais baseadas em x86 usavam a técnica de leitura antecipada da sequência de instruções, procurando prováveis pontos de falha da virtualização e substituindo estas instruções em tempo de execução [14], o que provocava uma grande perda de desempenho.

Apenas com o lançamento das tecnologias chamadas Vanderpool pela Intel e Pacifica pela AMD, a partir de 2005, passou-se a contar com uma arquitetura capaz de verdadeiramente virtualizar um computador x86. Foi introduzido um novo modo de operação, ativado pela instrução VMXON, e 11 novas instruções para que fosse possível a virtualização [12], por exemplo, VMLAUNCH e VMRESUME, responsáveis respectivamente por iniciar e encerrar a execução de uma máquina virtual.

Com a arquitetura atualmente disponível, é possível capturar todo o estado da máquina virtual em uma estrutura de dados chamada Virtual Machine Control State, dando mais poder ao VMM para conhecer o motivo da sua evocação e qual a situação da máquina virtual no momento da geração do *trap*, além de facilitar o retorno à execução normal do sistema *guest* após o processamento.

4. APLICAÇÕES DE VIRTUALIZAÇÃO

Apesar de compartilharem os conceitos gerais expostos anteriormente, na prática cada aplicação de virtualização possui características específicas que dependem das restrições da tecnologia utilizada e que levam a diferentes padrões de desempenho. Estas características serão comentadas nas seções a seguir.

4.1. Aplicações comerciais

Com todas as facilidades que a virtualização provê ao gerenciamento de *datacenters* comerciais, esta tecnologia tem sido encontrada com cada vez mais frequência nas plataformas disponibilzadas para implantação de aplicações de negócios. Entretanto, o desempenho desta categoria de softwares não é necessariamente idêntico ao percebido em um ambiete não-virtualizado.

Por meio do uso de *benchmarks* voltados para aplicações comerciais como o TPC-C, SpecWeb99 e Specjbb2005, é possível verificar [15] que certos perfis de sistemas estão mais aptos à execução em ambiente virtualizado que outros, comparando os resultados em máquinas com e sem virtualização. Servidores Web e servidores de Aplicações Java podem ser instalados em servidores virtuais sem grande impacto em desempenho, enquanto que não se recomenda a instalação de servidores de Banco de Dados transacionais (OLTP) porque fazendo acesso a dados em volumes relativamente pequenos, o desempenho é limitado pelo processador.

Recomenda-se fortemente o uso para aplicações de sistemas de suporte a decisão (DSS ou OLAP), pois são aplicações que consultam intensivamente os discos, deixando o gargalo para o I/O e apresentando 80% a 90% da performance em relação ao mesmo *benchmark* executado em ambiente sem virtualização.

4.2. Aplicações científicas

Usuários de aplicações científicas têm de lidar com grandes volumes de dados [1] e execução de sistemas com alto grau de paralelismo. Tendo em vista estas características, existe um grande potencial para a adoção de tecnologias relacionadas a virtualização.

A facilidade de replicação proporcionada pela adoção de máquinas virtuais permite que, configurando apenas uma vez a aplicação responsável pelo processamento a ser paralelizado, esta máquina possa ser replicada tantas vezes quantas forem necessárias. Algoritmos de processamento de grandes volumes de dados como o MapReduce podem ser usado nestes casos.

A virtualização também permite que, quando diversas aplicações científicas já existentes precisam ser usadas conjuntamente, cada uma tenha seu próprio ambiente de execução com seus requisitos originais de sistema operacional e recursos, dispensando-se migrações de plataforma, recompilações e outras potenciais dificuldades.

Um relato de uso [17] para aplicações de bioinformática indica que estes benefícios teóricos foram verificados na prática, dando ao pesquisador um ambiente computacional amigável, com certo grau de abstração em relação aos detalhes físicos e sem perda significativa de desempenho.

4.3. Mobile Virtualization

Dentro desta perspectiva o telefone móvel modular funcionará por meio de uma virtualização móvel. Atualmente, os programadores precisam reescrever cada aplicação, seja um jogo, um serviço de rede social, ou outros recursos, para cada um dos vários sistemas operacionais incluindo o Symbian da Microsoft Windows Mobile ou Google Android. Com a virtualização é possível a adição de recursos independentemente do sistema operacional em uso o que pode acelerar significativamente o processo de design dos aparelhos assim como a otimização dos seus recursos.

Normalmente para que as operações ocorram são necessários: um processador para comunicação, um para aplicações (e-mail) e outro para multimídia (gráficos, áudio e vídeo). Em um telefone virtualizado poderão ser apenas um ou dois processadores, em vez de três. As estimas de economia na fabricação de celulares pode alcançar a cifra de US $ 5 a US $ 10 por telefone [17].

4.4. Cloud Computing

O conceito de **computação em nuvem** refere-se à utilização da memória das capacidades de armazenamento e cálculo de computadores e servidores compartilhados e interligados por meio da internet, seguindo o princípio da computação em rede.

O armazenamento de dados é feito em serviços que podem ser acessados de qualquer lugar do mundo, a qualquer hora, não havendo necessidade de instalação de programas ou de armazenagem de dados. O acesso a programas, serviços e arquivos é remoto, através da Internet, daí a alusão à nuvem. O uso desse modelo/ambiente é mais viável do que o uso de unidades físicas.

Num sistema operacional disponível na Internet, a partir de qualquer computador e em qualquer lugar, pode-se ter acesso a informações, arquivos e programas num sistema único, independente de plataforma. O requisito mínimo é um computador compatível com os recursos disponíveis na internet. O PC torna-se apenas um *chip* ligado à Internet (agora a "grande nuvem" de computadores) sendo necessários somente os dispositivos de entrada teclado, mouse e saída (monitor).

Empresas como Google, IBM, Microsoft foram as primeiras a iniciar desenvolvimentos nessa "nuvem de informação" e, aos poucos, essa tecnologia vai deixando de ser utilizada apenas em laboratórios para ingressar nas empresas e em computadores domésticos, preferencialmente com códigos abertos.

Com relação à tipologia e também ao tipo de virtualização existem as seguintes divisões de cloud computing:

- **IaaS** - *Infrastructure as a Service* ou **Infra-estrutura como Serviço** : quando se utiliza uma porcentagem de um servidor, geralmente com configuração que se adeque à sua necessidade.
- **PaaS** - *Plataform as a Service* ou **Plataforma como Serviço:** utilizando-se apenas uma plataforma como um banco de dados, um web-service, etc.
- **DaaS** - *Development as a Service* ou **Desenvolvimento como Serviço**: as ferramentas de desenvolvimento tomam forma no *cloud computing* como ferramentas compartilhadas, ferramentas de desenvolvimento *web-based* e serviços baseados em mashup;
- **SaaS** - *Software as a Service* ou **Software como Serviço**: uso de um software em regime de utilização web (p.ex.: Google Docs);
- **CaaS** - *Communication as a Service* ou **Comunicação como Serviço**: uso de uma solução de Comunicação Unificada hospedada em Data Center do provedor ou fabricante.

Dependemos das necessidades das aplicações que serão implementadas, existem diferentes modelos, principalmente com relação à restrição ou abertura de acesso: ao negócio a ser acessado/compartilhado, ao tipo de informação e ao nível de visão desejados.

Os sistemas operacionais para Internet mais utilizados são: **Google Chrome OS**: (Google), todas as aplicações ou arquivos são salvos na nuvem e sincronizados na conta do Google, sem necessidade de salvá-los no computador; **Joli Os** (Tariq Krim), o ambiente de trabalho é chamado jolicloud, usa tanto aplicativos em nuvem quanto aplicativos ofline, baseado no ubuntu e esta sendo desenvolvido para funcionar no android. **YouOS**: (WebShaka), utiliza a linguagem Javascript para executar as operações. **DesktopTwo**: (Sapotek), tem como pré-requisito a presença do utilitário Flash Player para ser utilizado e desenvolvido em linguagem PHP. **G.ho.st**: (Global Hosted Operating SysTem) tem como diferencial em relação aos outros a possibilidade de integração com outros serviços como: Google Docs, Meebo, ThinkFree. **eyeOS**: (EyeOS Team) possui o código fonte aberto ao público, o objetivo é criar um ambiente com maior compatibilidade com os aplicativos atuais, MS-Office e OpenOffice, possui um abrangente conjunto de aplicativos, e o seu desenvolvimento é feito com linguagem PHP.

5. BENEFÍCIOS DA VIRTUALIZAÇÃO

Resumimos a seguir as principais vantagens apresentadas tanto na literatura quanto pelos fornecedores de sistemas virtualizados, que justificam a sua adoção mesmo tendo em vista a desvantagem de perda desempenho.

5.1. Segurança e Isolamento

Como é freqüente encontrar-se *bugs* em sistemas operacionais, por estes serem compostos por dezenas de milhões de linhas de código [10], a adoção de um modelo de isolamento entre processos composto por menos linhas de código (da ordem de milhares), como é o caso de um VMM, reduz a probabilidade de uma falha que possa colocar em risco a segurança de dados sensíveis ou aplicativos de missão crítica.

Os *drivers* de dispositivos, normalmente escritos e testados em modo usuário, mas executado em condições reais em modo *kernel*, também são um foco em potencial de falhas de sistemas. O uso de um *hypervisor* como camada de isolamento entre *drivers* e os demais elementos do Sistema Operacional pode melhorar a segurança e estabilidade de sistemas, de acordo com [3].

5.2. Eficiência e Confiabilidade

Colocando vários sistemas operacionais *guest* para executar em um único hardware físico, diferentes objetivos podem ser adotados, como minimizar o consumo de energia, maximizar o desempenho das máquinas virtuais ou maximizar a localidade, reduzindo o número de servidores.

Como uma máquina virtual pode ser transferida de um servidor físico para outro, podem ser aplicados algoritmos [6] para alocação dinâmica de forma mais eficiente, reduzindo a necessidade de um planejamento inicial que demanda muito conhecimento da carga real de trabalho, além de permitir maior flexibilidade à medida que as necessidades mudam.

Em um cluster com vários servidores, uma aplicação que tenha períodos distintos de alta demanda e de ociosidade pode ser mapeada, respectivamente, para uma máquina virtual que ocupa sozinha um hardware físico ou que o compartilha com outras VMs, permitindo assim um uso mais racional dos recursos e um melhor atendimento ao usuário final.

Também é possível manter registros instantâneos do estado de uma máquina virtual (*snapshots*) de modo que, no caso de uma falha em um dispositivo físico, uma instância praticamente idêntica a ela possa ser iniciada em outro hardware independente, garantindo a disponibilidade e confiabilidade nos serviços.

5.3. Suporte a Legados

A necessidade de manter em funcionamento aplicações que foram desenvolvidas para sistemas operacionais diferentes, muitos dos quais podem não ter mais suporte oficial, pode ser um empecilho ao gerenciamento de um *datacenter*, que leva à presença de hardware obsoleto para que estes sistemas continuem em uso. Com uma camada de virtualização, um sistema operacional como MS-DOS pode ser instalado como *guest* em uma máquina virtual dimensionada para prover um ambiente com as mesmas restrições.

De modo similar, aplicações que foram compiladas para serem executadas isoladamente em um servidor e para as quais não existe mais acesso ao código fonte, ou não há interesse em se investir em sua modificação, podem ser executadas em um servidor compartilhado através do isolamento proporcionado pela virtualização.

5.4. Treinamento e testes

Uma aplicação multiplataforma precisa ser testada em diversos sistemas operacionais antes de ser disponibilizada. Da mesma forma aplicações Web devem ter seu funcionamento e *layout* garantidos em várias versões de navegadores, também possivelmente rodando em sistemas operacionais diferentes. Todas as possíveis combinações entre navegadores, aplicativos e sistemas operacionais podem ser obtidas e mantidas em um único hardware físico, por meio de uma coleção de maquinas virtuais de teste.

Pelo mesmo princípio, treinamentos de instalação e manutenção de sistemas operacionais podem ser feitos por diversas turmas que compartilham o mesmo laboratório, pois cada aluno terá sua máquina virtual exclusiva, livre dos efeitos que um aluno de outra turma possa causar durante seus experimentos.

5.5. Compartilhamento de recursos

Processadores gráficos (GPUs) estão sendo cada vez mais usados como uma solução de hardware para acelerar aplicações computacionalmente intensivas. Geralmente são placas acopladas a um computador através de um barramento como, por exemplo, o PCI-Express e que são acessadas através de *drivers* ou diretamente pelo *kernel* do sistema operacional. Foi sugerido por [8] que este poder computacional pudesse ser compartilhado entre diversas máquinas virtuais através de uma camada adicional de hardware que expõe a GPU como um processador SIMD virtualizado.

6. TENDÊNCIAS E DESAFIOS

Tradicionalmente, o VMM centrou-se na partilha equitativa dos recursos do processador entre os domínios, com um imprevisível nível de qualidade em seu serviço. Contudo, com o rápido crescimento dos recursos de *hardware* e *software*, a avaliação de desempenho dos serviços de recursos do VMM está se tornando cada vez mais importante ou mesmo a chave para melhorar seu desempenho. De acordo com [7], uma das primeiras necessidades que devem ser abordadas futuramente para melhoria de sistemas virtualizados é a redução do número de *traps* necessárias para sua implementação, bem como o tempo para a sua execução.

Como as arquiteturas de CPU mais modernas não foram projetadas de forma "virtualizável", o desenvolvimento da tecnologia VMM é muito lento [7] sendo compreendido como um novo modo de execução a adição de tecnologias como o Vanderpool ao processador AMD, no caso da Intel, ou o Pacifica no caso dos processadores x86. O "novo" processador pode fazer com que haja uma execução direta de um VMM nas máquinas virtuais de forma segura e transparente melhorando o desempenho.

Desta forma a gestão de recursos é uma grande promessa como uma área para pesquisas futuras, no sentido de investigar formas de decisões de gestão cooperativa de recursos entre os VMMs e os sistemas operacionais convidados e, além disso, a gestão de recursos em todos os níveis centrais de dados.

O rápido crescimento dos recursos de hardware e software torna mais difícil e complicado a gestão dos VMMs criando um paradoxo entre a racionalização da utilização dos VMMs e o desempenho dos serviços de recursos em tempo real, isto tudo colocado dentro da perspectiva de armazenamento em nuvem, aplicação para uso intensivo de dados, organização em nó e escalabilidade.

A fim de agendar o recurso do sistema e melhorar a eficiência do seu desempenho serão necessárias construções estratégicas, interativas e em tempo real, entre múltiplas VMs em diferentes VMMs.

Em VMMs a segurança do sistema principal envolve a segurança do *host* e do hypervisor que uma vez comprometida quebra a segurança de todo o modelo. Assim, ataques contra o *hypervisor* serão os mais populares no reino dos "maliciosos atacantes", tornando a existência e a frequência dos *patches* imprescindíveis após a configuração do ambiente para garantir que o *hypervisor* esteja suficientemente seguro. Embora alguns métodos atuais de monitoramento de desempenho possam monitorar ou predizer o desempenho do VMM eles enfrentarão muitas dificuldades a partir do momento em que tiverem que monitorar ambientes de múltiplos VMMs dadas as diferenças entre ambos, tais como a estado de agendamento de recursos e os processos de processamento da tarefa. Para superar essas desvantagens, alguns novos modelos de avaliação de desempenho, de métodos de monitoramento, modelos de arquiteturas devem ser desenvolvidos para sistemas de múltiplos VMMs.

6.1. Gestão de Workload em Datacenter de VMMs

Virtualização tem o potencial de reduzir drasticamente o custo total dos centros de dados e aumentar a flexibilidade para implementações de *workloads*. Se as tendências atuais continuarem, o *datacenter* do futuro será em grande parte virtualizados, a plataforma base será composta de *hosts* físicos que rodam *hypervisors*, e os *workloads* serão executados dentro de uma plataforma VM.

A partir desta perspectiva [4] propõe a gestão de *workloads* no ambiente de *datacenters* virtualizados nas operações de *hosts* físicos próprios, ao contrário dos tradicionais *design* de datacenter centrados nas vantagens e desvantagens dos custos *versus* capacidade de execução requerida pelas aplicações dos usuários finais no datacenter. Assim a nova base de gestão seria projetar um datacenter virtualizado que atue sobre os fluxos de trabalho gerados pelos *workloads*.

A proposta de [4] obteve resultados satisfatórios dentro de testes realizados em dados reais de implementações virtualizadas em termos de avaliar o impacto da gestão dos fluxos de trabalho de um datacenter (exigências em I/O, rede, disco em dispositivos NAS).

Esta gestão produz diversas implicações para arquitetura dos computadores e consequentemente dos *datacenters*. Análises de custo/benefício precisam ser feitas em vários níveis. No de processadores, dado o aumento constante do uso core ou multi-core, caches maiores ou mesmo processadores mais antigos mais adaptados com extensões de virtualização. No nível de sistemas os requisitos de armazenamento e rede sugerem uma alta velocidade de barramento de I / O em conjunto com vários adaptadores necessários para as operações de gestão (migração ao

vivo, tolerância a falhas, alta disponibilidade). Desta forma, um tecido de convergência [20] pode ser uma solução, desde que os requisitos do armazenamento individual e de network não sobrecarreguem os adaptadores convergentes de nó ou de I/O.

6.2. Modelo de Arquitetura em Nuvem para armazenamento de aplicações com uso intensivo de dados

O armazenamento em nuvem pode fornecer alta escalabilidade, disponibilidade de dados, tolerância a falhas, segurança e boa relação de custo-benefício para os serviços de aplicações com uso intensivo de dados. Desta forma [1] propõe uma arquitetura em camadas para o armazenamento em nuvem composta por quatro camadas: para o usuário do aplicativo; para a plataforma de hospedagem dos aplicativos; e, para o gerenciamento e armazenamento de recursos. Também foi acrescentado um nó de gerenciamento de nós de I/O para que se possa obter um bom desempenho de I/O, problema persistente dado ao constante aumento de desempenho das CPUs e memórias, comparativamente ao desempenho de I/O. A configuração final da arquitetura é demonstrada na (Fig.5)

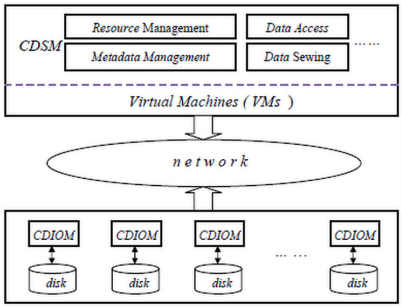

Figura 5: Organização do Subsistema de Armazenamento em Nuvem [1]

6.3 NoHype: Infraestrutura de Nuvem Virtualizada sem Virtualização

A tecnologia-chave subjacente nas infraestruturas em nuvem é a virtualização tanto que muitos a consideram uma das principais características ao invés de simplesmente um detalhe de implementação. Infelizmente, seu uso traz preocupações com a segurança já que executam múltiplas VMs ao mesmo tempo no servidor e como as camadas de virtualização executam um considerável papel nas operações VMMs, um ataque bem sucedido pode comprometer o controle da camada de virtualização, a confidencialidade e a integridade dos dados e dos *softwares*.

A remoção da camada *hypervisor*, mantendo as principais características habilitadas pela virtualização é proposta por [5] cujas principais características são: **uma máquina virtual (VM) por Core** - cada núcleo do processamento é dedicado e pode executar apenas uma única VM que não são compartilhados entre diferentes VMs o que atenua as ameaças de segurança relacionadas a canais laterais típicos no uso de recursos compartilhados (cache) ou em acessos a ambientes públicos hospedados, simplificando cobranças em termos de aplicações escaláveis; **particionamento forçado de memória** que garante a cada VM o acesso único e correto a memória física e ao intervalo que lhe foram atribuídos; **dispositivo virtual dedicado de I/O**: o dispositivo de modificações de I/O para poder suportar a virtualização permite que cada máquina virtual tenha acesso direto a um dispositivo virtual dedicado de I/O. As facilidades de gerenciamento de memória em conjunto com chipsets garantem que apenas as VMs autorizadas possam acessar a memória mapeada de I/O e somente em uma taxa limitada.

No NoHype o acesso a taxa limitada de cada barramento de I/O é alcançado através de um mecanismo de controle de fluxo onde o dispositivo de I/O controla a taxa de transmissão, cada dispositivo da VM é configurado com a taxa na qual ele pode ser acessado. Para os componentes de rede, as modificações implicam que os roteadores de Ethernet devem executar as funções de comutação e de segurança dos dados, e não uma opção em software na camada de virtualização.

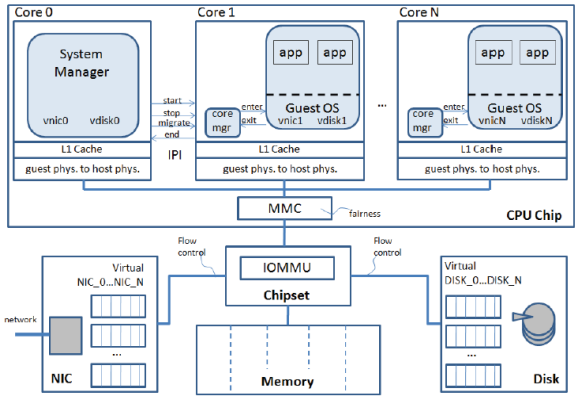

Figura 6: Sistema de Arquitetura No-Hype [5]

6.4 Padrões de Consumo de CPU e Memória cotados em Bolsa e Datacenter Verde:

O que vemos é que cada provedor tem criado a sua própria métrica, e as ofertas são baseadas em níveis de serviço, como VMs com quantidades pré-definidas de CPU, memória, rede, disponibilidade, ou ofertas por quantidade de storage, criando assim portifólios de ofertas semelhantes mais não exatos. Aventou-se a possibilidade de uma bolsa de CPU e memória, sendo cotada no mercado como se fosse arroz ou milho, mas o fato é que provedores de tecnologias de virtualização estudaram o caso e parece pouco provável que o mercado entre em um acordo de medida comum para consumo de CPU via internet, pois as ofertas de serviços já estão mudando.

Ofertas classificadas como PaaS, como é o caso da VMforce, oferecem uma nuvem onde o desenvolvedor ou empresa faz o *upload* de sua aplicação Java, ou até desenvolve direto no site. A aplicação é publicada, e o pagamento é feito por usuário que acessa. Recursos como processamento, memória, rede, storage e até banco de dados é alocado de forma dinâmica e transparente, o usuário do serviço não precisa nem planejar antes o quanto de recurso irá precisar. Se seu sistema online tiver um usuário, paga um valor; conforme o numero de visitas aumenta, o custo aumenta.

Nas nuvens privadas, inúmeros parâmetros estão sendo usados para gerenciamento do Datacenter virtualizado, chegando até ao ponto dos administradores usarem custo de TI *versus* Watt utilizado, amarrando assim a forma exata de investimento em energia elétrica contra o que pode ser produzido pelo departamento de TI com aquela quantidade de energia. Este modelo ajuda a administração de recursos de TI *versus* eletricidade, prove um modelo de custo exato, e facilita a cobrança interna de consumo de tecnologia por outras áreas.

Naturalmente este modelo foi criado como um dos pilares para alcançar o Datacenter Verde.

7. CONCLUSÃO

Na tecnologia virtual o sistema de computador pode agregar todos os tipos de recursos de dados, *software* e *hardware* para fornecer diferentes serviços e tarefas. Como a virtualização separa o *hardware* da gestão de *softwares* ela pode fornecer recursos úteis tais como o isolamento de desempenho, a consolidação de servidores, a migração em tempo real, os ambientes portáteis para os modernos sistemas de computação e a tecnologia de segurança, conforme descrito nas seções anteriores. Com a alteração das plataformas de mainframe para desktop e o surgimento de novas demandas provocadas por *cloud computing* e portabilidade – de acesso, aplicativos, sistemas operacionais – a virtualização assume condição *sine qua non* para a pesquisa em computação.

8. REFERÊNCIAS

[1] Huo, Yanmei; Wang, Hongyuan; Hu, Liang; Yang, Hongji; , "A Cloud Storage Architecture Model for Data-Intensive Applications," *Computer and Management (CAMAN), 2011 International Conference on* , vol., no., pp.1-4, 19-21 May 2011 doi: 10.1109/CAMAN.2011.5778817

[2] Hendricks, E. C.; Hartmann, T. C.; , "Evolution of a virtual machine subsystem," IBM Systems Journal , vol.18, no.1, pp.111-142, 1979 doi: 10.1147/sj.181.0111

[3] Borghei, E.; Azmi, R.; Ghahremanian, A.; Nemati, H.; , "Virtual machine based security architecture" Internet Security (WorldCIS), 2011 World Congress on , vol., no., pp.210-215, 21-23 Feb. 2011.

[4] Vijayaraghavan Soundararajan and Jennifer M. Anderson. 2010. "The impact of management operations on the virtualized datacenter". In Proceedings of the 37th annual international symposium on Computer architecture (ISCA '10). ACM, New York, NY, USA, 326-337.

[5] Eric Keller, Jakub Szefer, Jennifer Rexford, and Ruby B. Lee. 2010. "NoHype: virtualized cloud infrastructure without the virtualization". In Proceedings of the 37th annual international symposium on Computer architecture (ISCA '10). ACM, New York, NY, USA, 350-361.

[6] Kleineweber, C.; Keller, A.; Niehorster, O.; Brinkmann, A.; , "Rule-Based Mapping of Virtual Machines in Clouds," Parallel, Distributed and Network-Based Processing (PDP), 2011 19th Euromicro International Conference on , vol., no., pp.527-534, 9-11 Feb. 2011 doi: 10.1109/PDP.2011.69

[7] Yunfa Li; Wanqing Li; Congfeng Jiang; , "A Survey of Virtual Machine System: Current Technology and Future Trends," Electronic Commerce and Security (ISECS), 2010 Third International Symposium on , vol., no., pp.332-336, 29-31 July 2010. doi: 10.1109/ISECS.2010.80

[8] José Duato, Francisco D. Igual, Rafael Mayo, Antonio J. Peña, Enrique S. Quintana-Ortí and Federico Silla; "An Efficient Implementation of GPU Virtualization in High Performance Clusters". Euro-Par 2009 – Parallel Processing Workshops. Lecture Notes in Computer Science, 2010, Volume 6043/2010, 385-394.

[9] Amit Vasudevan, Jonathan M. McCune, Ning Qu, Leendert van Doorn and Adrian Perrig. "Requirements for an Integrity-Protected Hypervisor on the x86 Hardware Virtualized Architecture", TRUST 2010, Lecture Notes in Computer Sciences 6101, pp. 141–165, 2010. Springer-Verlag Berlin Heidelberg 2010.

[10] Gerald J. Popek and Robert P. Goldberg. 1974. Formal requirements for virtualizable third generation architectures. Communications of ACM, July 1974, Volume 17, Number 7.

[11] Loic Duflot, Olivier Grumelard, Olivier Levillain, and Benjamin Morin. "On the Limits of Hypervisor- and Virtual Machine Monitor-Based Isolation". Towards Hardware-Intrinsic Security, Information Security and Cryptography, DOI 10.1007/978-3-642-14452-3_16, C Springer-Verlag Berlin Heidelberg 2010.

[12] John L. Hennessy e David A. Patterson. Arquitetura de computadores - Uma abordagem quantitativa. Quarta Edição, 2008. Editora Campus.

[13] David Marshall, Wade A. Reynolds, and Dave McCrory. Advanced server virtualization [recurso eletrônico] : VMware and Microsoft platforms in the virtual data center / by Taylor & Francis Group, LLC, 2006.

[14] John Scott Robin and Cynthia E. Irvine. Analysis of the Intel Pentium's Ability to Support a Secure Virtual Machine Monitor. Proceedings of the 9th USENIX Security Symposium Denver, Colorado, USA August 14 –17, 2000

[15] Tsuyoshi Tanaka, Toshiaki Tarui, and Ken Naono. "Investigating Suitability for Server Virtualization using Business application Benchmarks". Proc. VTDC'09, June 15, 2009, Barcelona, Spain.

[16] Olga Kharif, "Virtualization Goes Mobile". Bloomberg Businessweek Online. April 22, 2008. Available online at http://www.businessweek.com/technology/content/apr2008/tc20080421_235517.htm

[17] Andréa Matsunaga, Maurício Tsugawa, José Fortes. "CloudBlast: combining MapReduce and Virtualization on Distributed Resources for Bioinformatics Applications". Fourth IEEE International Conference on eScience, 2008.

[18] Smith, J.E.; Ravi Nair; "The architecture of virtual machines" *IEEE Computer* Volume: 38 , Issue: 5
Digital Object Identifier: 10.1109/MC.2005.173
Publication Year: 2005 , Page(s): 32 – 38.

[19] Marcos Aurelio Pchek Laureano. Máquinas virtuais e Emuladores – Conceitos, técnicas e aplicações. Editora Novatec, 2006.

[20] H. Andrés Lagar-Cavilla, Joseph A. Whitney, Adin Scannell, Philip Patchin, Stephen M. Rumble, Eyal de Lara, Michael Brudno, M. Satyanarayanan; "SnowFlock: Rapid Virtual Machine Cloning for Cloud Computing". In Proceedings of the 4th ACM European conference on Computer systems (EuroSys '09). ACM, New York, NY, USA, 1-12.