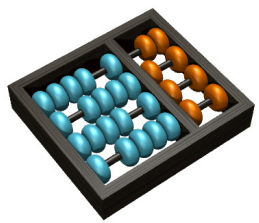
Uma Visão Geral Sobre Sistemas Virtualizados

Rodolfo Wottrich, Thiago Genez e Walisson Pereira

MO401 – Arquitetura de Computadores

13 de Junho de 2012



Cronograma

1. Introdução
   - Motivação
2. Conceitos básicos
3. Virtualização em x86
4. Vantagens e desvantagens de sistemas virtualizados
5. Tecnologias de virtualização
6. Tendências
7. Considerações finais

UNICAMP

UNICAMP

Motivação

- A motivação desta apresentação é oferecer uma visão geral dos mais importantes aspectos relacionados ao desenvolvimento e utilização de sistemas computacionais virtualizados.

UNICAMP

Objetivo

- Virtualização é uma camada de software que abstrai as características físicas do hardware, fornecendo recursos virtualizados para as aplicações de alto nível.
- Máquina real é logicamente dividida em:
  - *Virtual Machine Monitor* (VMM) ou *hypervisor*
  - *Virtual Machine*

UNICAMP

Histórico I

- **anos 60**:
  - A IBM, usando o IBM 7044, fez uma imagem do mesmo para cada partição do sistema para compreender melhor sistemas multiprogramados.
  - A IBM construiu o sistema operacional (SO) chamado *Cambridge Monitoring System* (CMS) para sistemas *time sharing*.
- **anos 70**:
  - O conceito de máquinas virtuais (VM) é formalizado.
- **anos 80**:
  - O barateamento dos PCs reduziu o interesse pelos sistemas virtualizados.
- **anos 90 em diante**:
  - A linguagem de programação Java é lançada.
  - Popularização dos sistemas virtualizados.

UNICAMP

UNICAMP

Abstração e Virtualização I

- **Abstração** consiste em providenciar uma interface simplificada e homogênea para acessar os recursos (hardware ou software)
- **Virtualização** proporciona uma camada de compatibilidade entre:
  - software e hardware
  - entre software

UNICAMP

Interfaces de Sistema I

- O Sistema Operacional oferece uma visão abstrata dos recursos de hardware, a fim de facilitar o uso e desacoplar dependências tecnológicas subjacentes.
- As interfaces mais utilizadas entre os componentes de computação são:
  - Arquitetura do conjunto de instruções – *Instruction Set Architecture* (ISA)
  - Chamadas de sistema
  - Biblioteca de sistema

UNICAMP

Interfaces de Sistema II

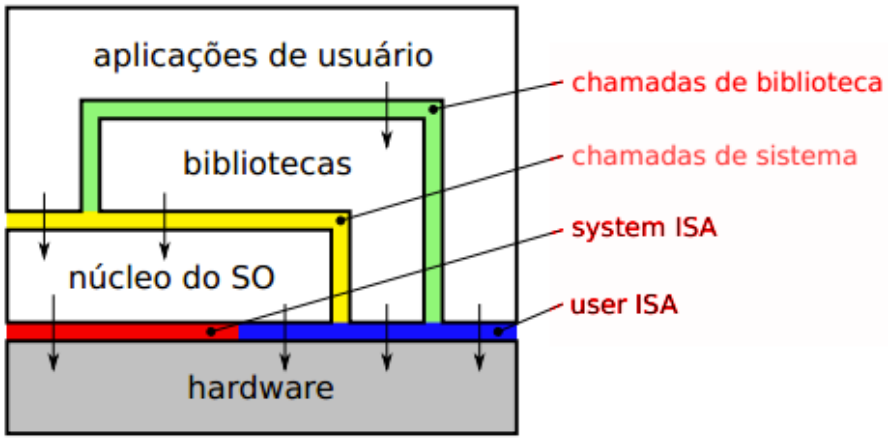


Figura 1: Componentes e interfaces de um SO [Car08].

Interfaces de Sistema III

- **Arquitetura do conjunto de instruções**: Conjunto de instruções, ou código de máquina, aceito pelo processador.
- Dividido em duas partes:
  - **instruções de usuário**: instruções não-privilegiadas do processador e não oferecem risco ao SO.
  - **instruções de sistema**: instruções privilegiadas do processador e devem ser executadas pelo núclo do SO ou sob sua supervisão.

UNICAMP

Interfaces de Sistema IV

- **Chamadas de sistema**: mecanismo utilizado pelos processos para solicitar um serviço ao SO.
- **Chamadas de biblioteca**: coleção de subprogramas que encapsulam chamadas de um SO, facilitando o desenvolvimento de aplicações para o próprio SO.

Virtualização e Máquinas Virtuais I

- Normalmente as aplicações escritas para uma certa plataforma operacional não funciona em outras plataformas.
- Incompatibilidade do hardware (ISA) e do software (*libcalls*)
- Virtualização é diferente de abstração!!!

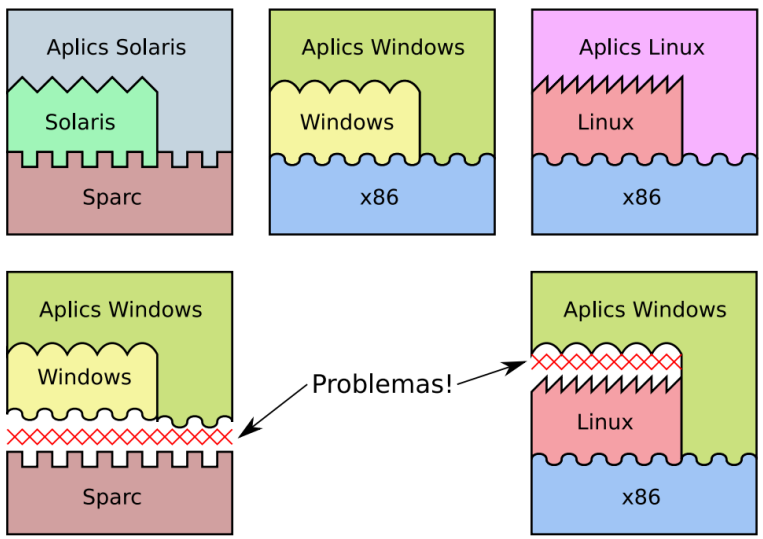


Figura 2: Problemas de compatibilidade entre interfaces [Lau06].

Virtualização e Máquinas Virtuais II

- Solução da incompatibilidade entre plataformas operacionais:
  - Tecnologia de *virtualização*.
  - O monitor da máquina virtual (VMM) ou *hypervisor*:
    - Responsável pelo controle e virtualização dos recursos físicos compartilhados entre as máquinas virtuais.
- *"A máquina virtual é definida como uma cópia eficiente e isolada de uma máquina real"* [PG74].

Figura 3: Acoplamento entre interfaces distintas [Lau06].

Tipos de máquinas virtuais

- De acordo com o sistema convidado, as máquinas virtuais são classificadas em 2 tipos:
  - **VM de Aplicação**
    - Suportam apenas uma aplicação específica
    - Java Virtual Machine (JVM)
    - Código Fonte (.java) → *bytecode* → JVM → binário (arq. específica)
    - Aplicativos em Java são independentes da plataforma operacional
  - **VM de Sistema**
    - Suportam sistemas operacionais completos
    - Aplicativos e serviços do SO
    - Exemplo: *VMware* e *Virtual Box*

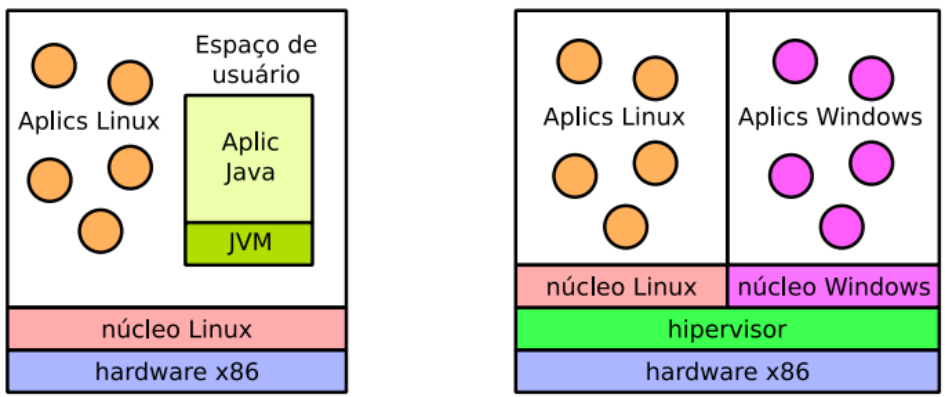


Figura 4: VM de aplicação (esq.) e de sistema (dir.) [LM08].

Tipos de *hypervisor* I

- Há duas abordagens de *hypervisor* de sistema
  - **Nativos (tipo I)**
    - Implementado diretamente no hardware real
    - Sem um sistema operacional subjacente
  - **Convidados (tipo II)**
    - Executado como um processo normal de um SO nativo subjacente
    - Cada *hypervisor* suporta, de modo geral, 1 VM
    - Mais VMs $\rightarrow$ mais processos *hypervisors* no SO nativo
  - *Hypervisor* nativo maior desempenho
    - Menos intermediário entre máquina real e a virtual
    - Mais complexo
  - Utiliza-se técnicas de emulação ou simulação

UNICAMP

Tipos de *hypervisor* II

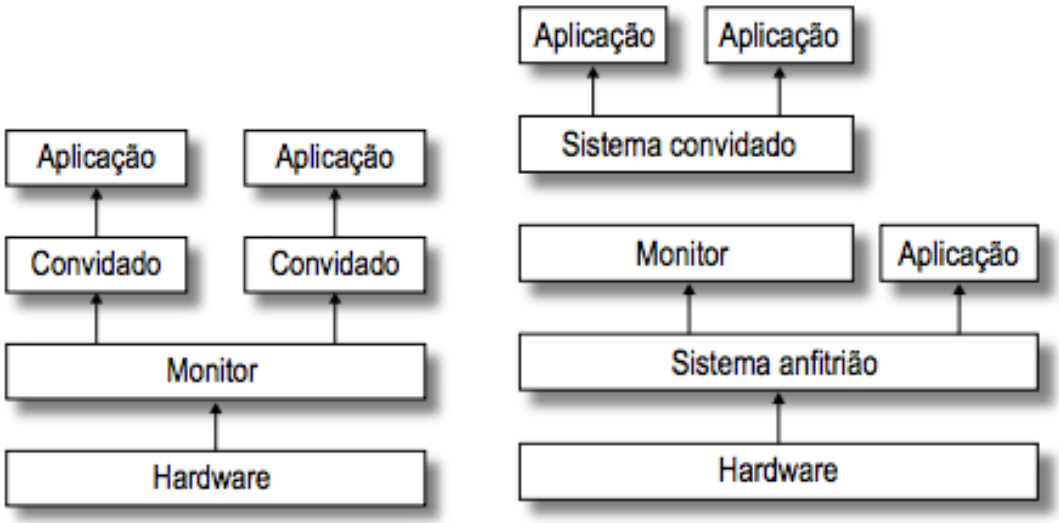


Figura 5: *Hypervisor* nativo (esq.) e convidado (dir.) [LM08].

- Abordagens teóricas
- Prática: otimiza o desempenho do *hypervisor* para melhorar o acesso ao hardware real. Exemplo: Paravirtualização

UNICAMP

Anéis de proteção da CPU I

- Anéis de Proteção ou Domínio de Proteção
  - Controla acesso conjunto de instruções
  - Controla acesso aos recursos compartilhados
  - Níveis de privilégios para proteger o hardware (e periféricos)
  - Anel 0 geralmente é executado pelo SO (sistema sem virtualização)
  - Quanto mais distante um anel se encontra do central, menor será o seu nível de privilégio
  - Arquitetura *x*86: 4 anéis
    - Anel 0 (espaço de *kernel*)
    - Anel 3 (espaço de usuário)

UNICAMP

Anéis de proteção da CPU II

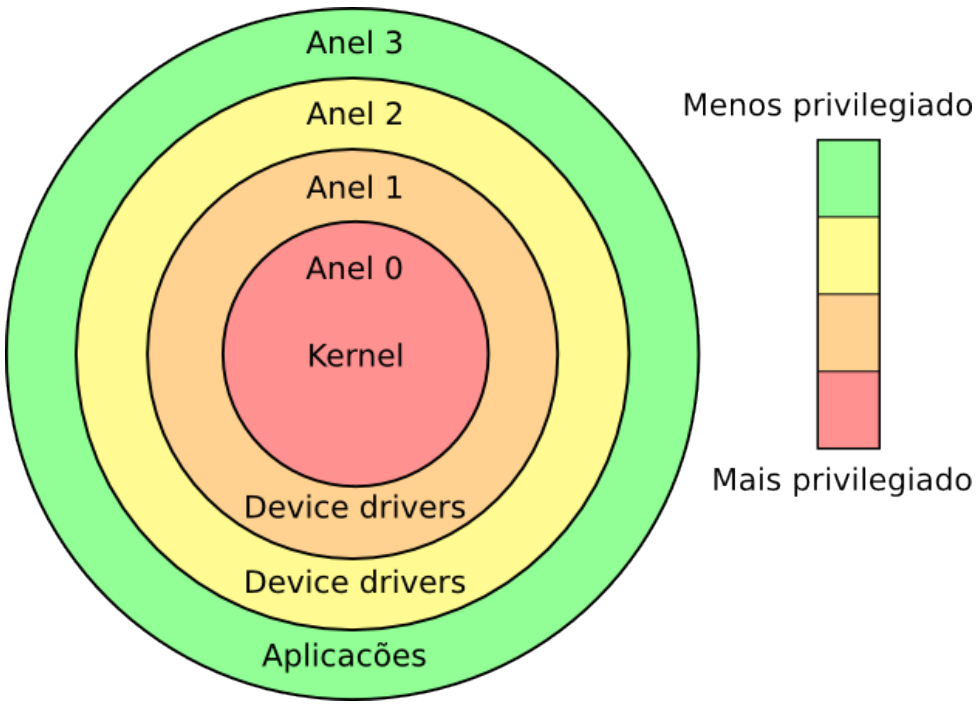


Figura 6: Anéis de privilégios da arquitetura *x*86

Anéis de proteção da CPU III

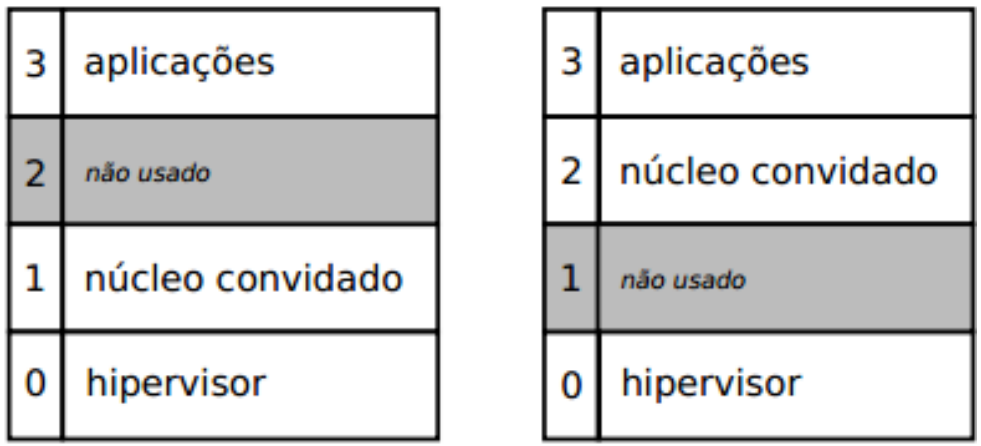


Figura 7: Virtualização com modelo 0/1/3 (esq.) e 0/2/3 (dir.) [Car08]

Teorema de Popek & Goldberg I

- Em 1974 Popek & Goldberg descreveram em [PG74] um teorema:
- *"Para qualquer computador convencional de terceira geração, um hipervisor pode ser construído se o conjunto de instruções sensíveis daquele computador for um subconjunto de seu conjunto de instruções privilegiadas."*
- **Computador convencional de 3ªgeração**
  - Qualquer computador seguindo a arquitetura de *Von Neumann*, que suporte memória virtual e dois modos de operação:
  - *modo usuário* (anel 3) e *modo privilegiado* (anel 0)

UNICAMP

Teorema de Popek & Goldberg II

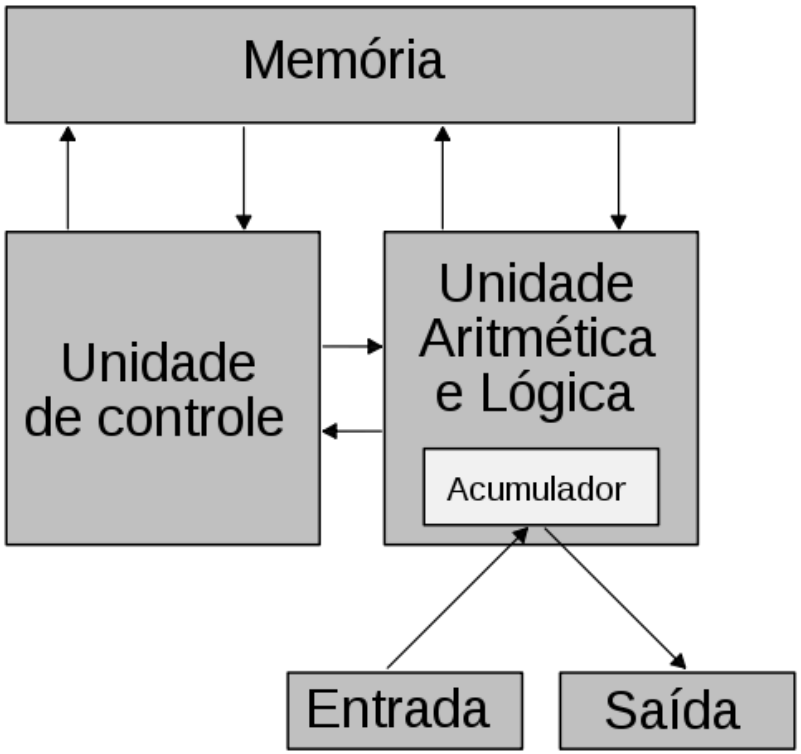


Figura 8: Arquitetura de von Neumann

Teorema de Popek & Goldberg III

- **Instruções sensíveis**
  - Podem consultar ou alterar o status do processador, ou seja, os registradores que armazenam o status atual da execução no hardware
- **Intruções privilegiadas**
  - Execução em modo usuário gera uma exceção, ou seja, é interrompida e o sistema operacional é notificado.
  - São acessíveis somente por meio de códigos executando em nível privilegiado (código de núcleo).

UNICAMP

Teorema de Popek & Goldberg IV

- Segundo o teorema:
  - Toda instrução sensível deve ser também privilegiada
  - Quando uma instrução sensível for executada por uma aplicação não-privilegiada (núcleo do SO convidado, anel 1 ou 2) irá provocar uma interrupção que deverá ser interceptada e tratada pelo *hypervisor*
  - *Hypervisor* irá emular o efeito desejado da instrução sensível
  - Garantindo o **isolamento** entre VMs
  - Instruções não-privilegiadas podem ser executadas diretamente no hardware real $\rightarrow$ não-prejudiciais à virtualização
- Arquitetura *x*86, a princípio, não obedecia este teorema
  - Instruções sensíveis que executem sem gerar interrupções
  - *Hypervisor* não consegue interceptá-las e interpretá-las

UNICAMP

Formas de Virtualização

- Virtualização completa ou total
- Paravirtualização

UNICAMP

Virtualização completa ou total

- *Hypervisor* situado no anel 0
- Disponibiliza uma réplica do hardware subjacente
- Alguns autores chamam de *virtualização do hardware*
- Toda interface (ISA) é virtualizada
- Vantagem:
  - SO visitante não precisa sofrer modificações para rodar na VM
  - *Tradução binária*: binário SO $\rightarrow$ *hypervisor* $\rightarrow$ binário hardware
    - Em tempo de execução
    - Detectar e tratar instruções sensíveis não-privilegiadas que não geram interrupções ao serem invocadas pelo SO *guest* $\rightarrow$ problema na *x*86
- Desvantagem:
  - Queda do desempenho em até 30%
- Exemplos: *VMware*, *VirtualBox* e *QEmu*

UNICAMP

Paravirtualização

- SO convidado tem "consciência" que está sendo virtualizado
- SO convidado precisa ser modificado
  - Adaptação é feita no *kernel*
- Tradução binária não é realizada
  - Aumenta o desempenho da virtualização
  - Precisa verificar apenas interface de sistema (instruções sensíveis)
  - Interface de usuário é mantida (apps do usuário sem modificação)
- Vantagem:
  - Acesso ao hardware diretamente (não-privilegiadas)
  - Monitorado pelo *hypervisor*, que informa os limites (*e.g.* disco e memória)
- Desvantagem:
  - Alteração do *kernel*
  - Linux: modificar 2995 linhas de código ($\approx 1,36\%$ do código fonte)
- Exemplo: *Xen*

Virtualização total e Paravirtualização

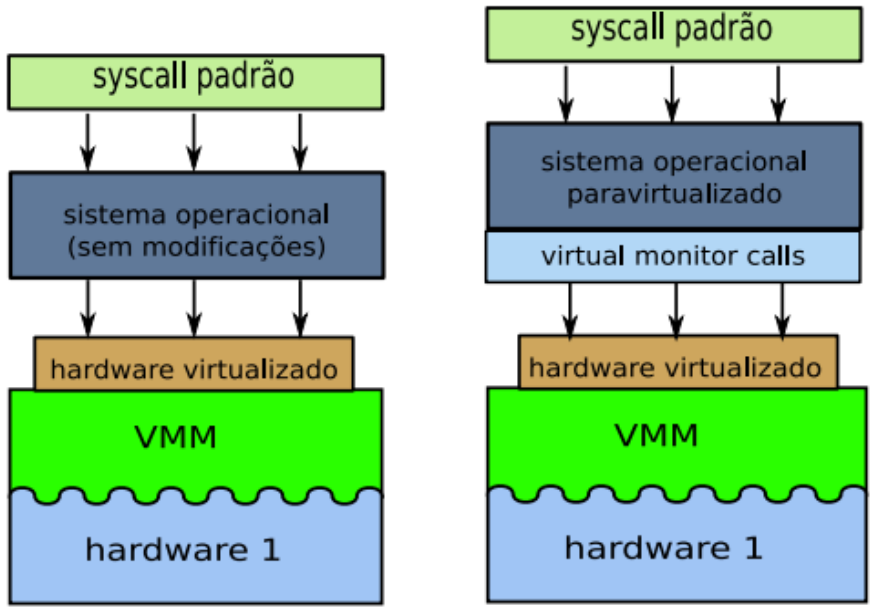


Figura 9: Virtualização total (esq.) e paravirtualização (dir.) [LM08]

UNICAMP

Virtualização em *x86*

- Arquitetura *x86* não foi projetada, a princípio, para ser virtualizada
- 17 instruções sensíveis que não são privilegiadas cuja execução em modo usuário não gera uma exceção, mas é tratada pela própria *x86*
- Viola o teorema de Popek & Goldberg
- Problema:
  - *Hypervisor* executa no anel 0 (modo privilegiado) → gerencia recursos
  - SO único software que deve executar no anel 0 → gerencia recursos
  - *x86* for virtualizada → anel 1 ou 2
  - Virtualização é transparente → a *x86* "pensa" que está no anel 0
  - Executar uma das 17 instruções "críticas" → abortadas
  - Ideal: Gerar uma exceção e ser tratada pelo *hypervisor*
  - Desafio: lidar com essas 17 instruções críticas
  - Virtualização completa → tradução binária (queda no desempenho)
  - Solução: *virtualização assistida pelo hardware*
- Intel-VT (*Vanderpool*) em 2005 e AMD-V (*pacífica*) em 2006

UNICAMP

Virtualização assistida pelo hardware I

- Obedece o Teorema de Popek & Goldberg
- Divide o processador *x*86 em 2 modos de operação:
  - *root* e *non-root*
- Modo *root*:
  - Destinado à execução do *hypervisor*
  - Equivale ao funcionamento do processador convencional
- Modo *non-root*
  - Destinado à execução das VMs
- São definidos 2 modos de operação:
  - VM *entry* (transição *root* $\rightarrow$ *non-root*)
  - VM *exit* (transição *non-root* $\rightarrow$ *root*).
- É definido a estrutura *Virtual-Machine Control Structure* (VMCS)
  - Gerenciar o estado do processador (conteúdo dos registradores)
  - 2 áreas: uma para os SOs convidados e outra para o *hypervisor*

UNICAMP

Virtualização assistida pelo hardware II

- Funcionamento:
  - **VM entry**: o estado do processador é salvo na área de *hypervisor* da VMCS e, em seguida, o estado do processador é lido a partir da área de SOs convidados da VMCS
  - **VM exit**: o estado do processador é salvo na área de SOs convidados e o estado anterior do *hypervisor* é restaurado.
  - Ambas transições são geranciadas pelo *hypervisor*
- As instruções sensíveis e as interrupções (geradas dentro da VM no modo *non-root*) provocam a transição VM *exit*, devolvendo o processador ao *hypervisor* em modo *root*
- *Ring deprivileging*:
  - SO opera em um anel 0 fictício
  - Não tem conhecimento da presença do *hypervisor* em um nível de privilégio superior

UNICAMP

Virtualização assistida pelo hardware III

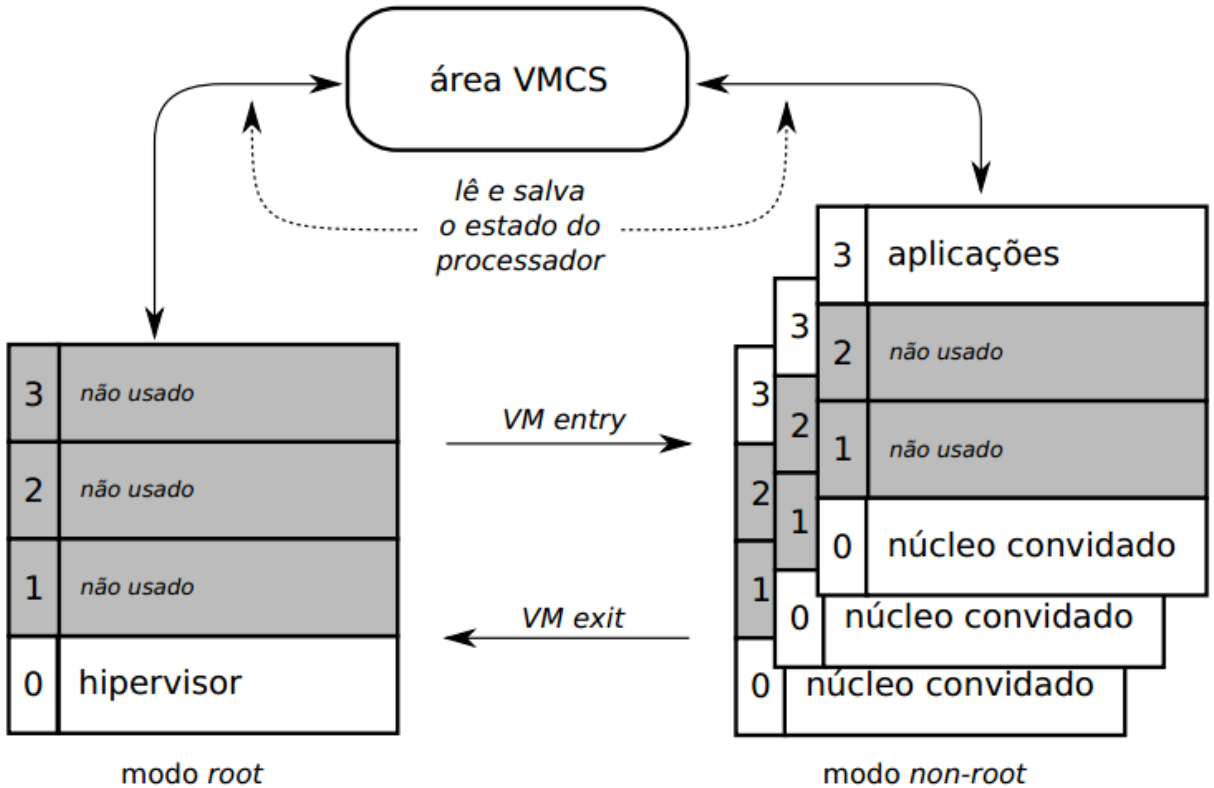


Figura 10: Visão geral da arquitetura Intel VT [Car08]

UNICAMP

Vantagens de sistemas virtualizados

- Flexibilidade de alocação de recursos
- Disponibilidade
- Escalabilidade
- Uso eficiente do hardware
- Segurança
- Custo
- Aplicações legadas

Desvantagens de sistemas virtualizados

- Sobrecarga
- SPOF (*Single point of failure*)
- A interface de administração

UNICAMP

Xen

- É baseado na paravirtualização.
- O Xen não possui *drivers* de dispositivos e, por isso, não é possível rodar um SO convidado diretamente nele.
- É necessário que o SO convidado tenha ciência de que está sendo operado sobre o núcleo Xen.
- Uma VM é instanciada no *domínio* 0 e faz a comunicação entre o Xen e os SO hóspedes.
- O VM no *domínio* 0 possui privilégios para acessar os dispositivos de E/S e as demais VM (*domínio U*).
- A VM *domínio* 0 tem acesso direto aos recursos da máquina física, enquanto as demais têm acesso a uma abstração dos recursos.

UNICAMP

Kernel-based Virtual Machine – KVM

- É uma solução de virtualização que se aproveita de toda estrutura de *drivers* já existente no *kernel* para ter acesso ao hardware.
- Necessita de um processador com suporte a virtualização (Intel VT ou AMD-V).
- Está contido no *kernel* do *Linux* desde a versão 2.6.20.

UNICAMP

VirtualBox

- Emula completamente o hardware.
- Não necessita de um processador com suporte a virtualização para executar uma máquina virtual.
- Caso o processador possua suporte a virtualização, o VirtualBox é capaz de se aproveitar disso.
- Possui uma interface intuitiva.

UNICAMP

UNICAMP

Provedores de Infraestrutura I

- Gerenciamento de *datacenters*:
  - Instanciação de VMs
  - Mapeamento dinâmico dos recursos físicos
  - Modelo *Infrastructure-as-a-Service* (IaaS) de computação em nuvem
  - Tarifação *pago-pelo-uso*

Provedores de Infraestrutura II

- Desafio: *hot migration*
  - Migrar VMs entre máquinas físicas do *datacenter*
  - Possibilita alocações de recursos para manutenção, upgrades, etc.
  - Gera novos workflows no gerenciamento do *datacenter*
  - Computação verde: economia de energia, preservação do meio ambiente
  - Introduz novas métricas de tarifação (energia *vs.* desempenho)

UNICAMP

Segurança

- Servidores virtuais: sujeitos aos mesmos ataques que antigem os físicos
- Cada vez mais ameaças explorando falhas no *hypervisor*
  - *Hypervisor* exposto a ameaças: VMs expostas também
- Segundo [LLJ10]: segurança total do sistema é baseada na segurança tanto do *hypervisor* quanto do SO convidado
- Se a segurança de qualquer elemento do sistema estiver comprometida, o sistema como um todo é falho
- Tendência: melhores *hypervisors* serão aqueles com atualizações constantes contra ameaças conhecidas e emergentes

UNICAMP

UNICAMP

Considerações Finais

- Virtualização: ferramenta já indispensável ao estado atual da computação
  - Adaptação de recursos computacionais a novas necessidades
  - Uso mais racional/controlado de recursos físicos
- Não é ideia recente, mas é em época recente que ganhou importância
  - Advento dos grandes *datacenters* e da computação em nuvem

UNICAMP

Obrigado!
Questões?

rodolfo.wottrich@students.ic.unicamp.br
thiagogenez@ic.unicamp.br
walisson@ic.unicamp.br

UNICAMP

Referências Bibliográficas I

Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield.

Xen and the art of virtualization.

In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, pages 164–177, New York, NY, USA, 2003. ACM.

Alexandre Carissimi.

Virtualização: da teoria a soluções.

In *Minicursos do Simpósio Brasileiro de Redes de Computadores (SBRC'08)*, pages 173–207, 2008.

Marcos Aurelio Pchek Laureano.

Máquinas Virtuais e Emuladores: Conceitos, Técnicas e Aplicações.

Novatec, 2006.

Referências Bibliográficas II

Qiang Li, Qinfen Hao, Limin Xiao, and Zhoujun Li.

Adaptive management of virtualized resources in cloud computing using feedback control.

In *1st International Conference on Information Science and Engineering (ICISE 2009)*, pages 99 –102, 26-28 2009.

Yunfa Li, Wanqing Li, and Congfeng Jiang.

A survey of virtual machine system: Current technology and future trends.

In *2010 Third International Symposium on Electronic Commerce and Security (ISECS'10)*, pages 332 –336, july 2010.

Marcos Aurelio Pchek Laureano and Carlos Alberto Maziero.

Virtualização: Conceitos e aplicações em segurança.

In *VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas*, pages 139–187, 2008.

UNICAMP

Referências Bibliográficas III

Richard McDougall and Jennifer Anderson.

Virtualization performance: perspectives and challenges ahead.

SIGOPS Oper. Syst. Rev., 44(4):40–56, dec 2010.

Gerald J. Popek and Robert P. Goldberg.

Formal requirements for virtualizable third generation architectures.

Commun. ACM, 17(7):412–421, jul 1974.

John Scott Robin and Cynthia E. Irvine.

Analysis of the intel pentium's ability to support a secure virtual machine monitor.

In *Proceedings of the 9th conference on USENIX Security Symposium - Volume 9*, SSYM'00, pages 10–10, Berkeley, CA, USA, 2000. USENIX Association.

UNICAMP

Referências Bibliográficas IV

J. Sahoo, S. Mohapatra, and R. Lath.

Virtualization: A survey on concepts, taxonomy and associated security issues.

In *2010 Second International Conference on Computer and Network Technology (ICCNT'10)*, pages 222 –226, april 2010.

James E. Smith and Ravi Nair.

The architecture of virtual machines.

Computer, 38(5):32–38, may 2005.

UNICAMP

Referências Bibliográficas V

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.

In *Proceedings of the 3rd international conference on Trust and trustworthy computing*, TRUST'10, pages 141–165, Berlin, Heidelberg, 2010. Springer-Verlag.

UNICAMP