Iniciar apresentação

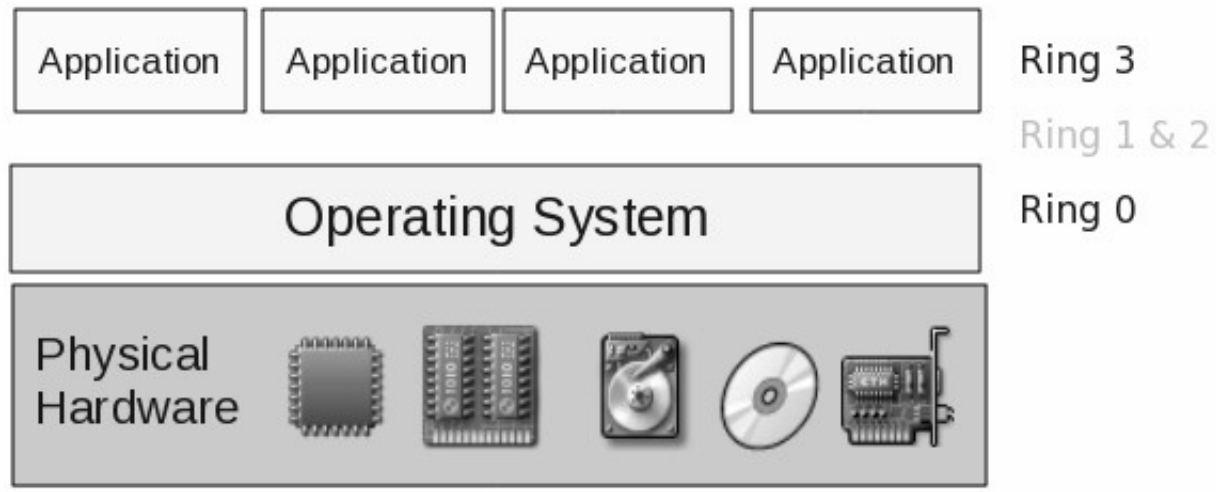
Aula 02 - Conhecendo as soluções livres de virtualização

Sobre esta aula

- Linha do tempo das soluções de virtualização livres.
- Evolução tecnológica das técnicas de virtualização.
- Distribuições, versões, aplicativos.

A arquitetura x86/x86_64

- A fim de proporcionar um ambiente seguro para um sistema operacional, a arquitetura x86 fornece um mecanismo para isolar aplicações de usuários do sistema operacional usando a noção de níveis de privilégio.
- O processador fornece 4 níveis de privilégio, também conhecidos como rings (anéis) que são dispostos em uma forma hierárquica, de 0 a 3. O ring 0 é o mais privilegiado, com acesso total ao hardware e é capaz de executar instruções privilegiadas.

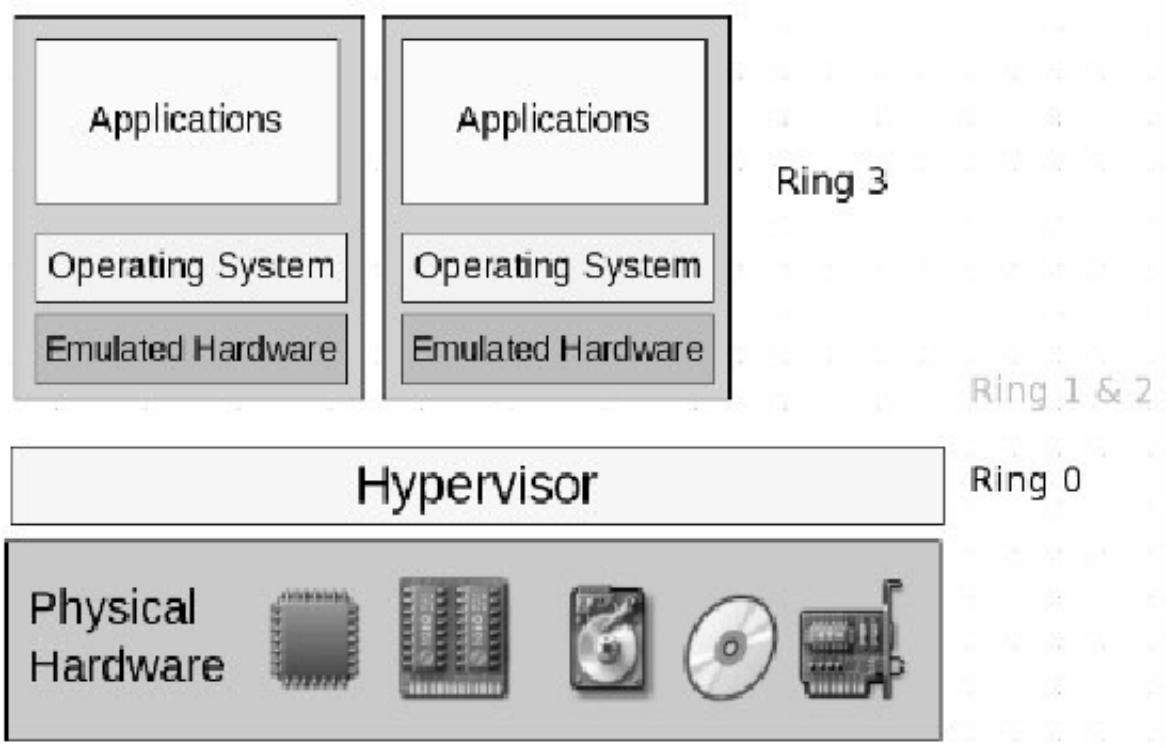


- O sistema operacional roda no ring 0 com o seu kernel controlando o acesso ao hardware.
- Os rings 1, 2 e 3 funcionam em um nível mais baixo de privilégios e são impedidos de executar instruções reservadas para o sistema operacional.
- Sistemas operacionais comumente utilizados, tais como Linux e Windows da Microsoft funcionam em ring 0 e programas de usuário rodam em ring 3. Rings 1 e 2 historicamente não tem sido usados.

A arquitetura x86/x86_64

- Embora este modelo proporcione benefícios para as tradicionais implementações "bare metal" de SOs, ela apresenta desafios em um ambiente virtualizado. Em um ambiente virtualizado o hypervisor deve ser executado no nível mais privilegiado, controlando todo o hardware. Neste modelo, as máquinas virtuais são executadas em um ring inferior desprivilegiado, geralmente no ring 3.
- Dentro do ring 3, podemos ver a máquina virtual em execução com um sistema operacional rodando no hardware virtual (emulado). Como o sistema operacional foi originalmente concebido para ser executado diretamente no hardware e espera ser executado no anel 0, ele vai fazer chamadas que são privilegiados e não autorizadas em ring 3.

A arquitetura x86/x86_64



A arquitetura x86/x86_64

- Quando o sistema operacional faz essas chamadas privilegiadas, o hardware irá interceptar as instruções e emitir uma falha, o que poderá destruir a máquina virtual.
- Grande parte do trabalho realizado antigamente em soluções de virtualização x86 é centrada em torno do manejo de remoção dos privilégios do sistema operacional em execução na máquina virtual, movendo o kernel do sistema operacional do anel de 0 para a 1 (ou superior).

- Os primeiros hypervisors, como o Bochs, criavam um sistema totalmente emulado, com um processador x86 e periféricos em software. Esta técnica resulta em um desempenho muito pobre, então técnicas mais avançadas.

No começo, havia a VMware

- 1998: VMware começa a trabalhar em um Virtual Machine Monitor (VMM) para arquitetura x86
    - Aplica a seguinte patente: Virtualization system including a virtual machine monitor for a computer with a segmented architecture
- 1999: VMware aparece para o mercado e lança o VMware Workstation 1.0 para Windows e Linux.
- 2001: VMware ESX e GSX Server 1.0.
- 2003: VMware ESX 2.0 - Suporte a SMP virtual e VMotion.

Tradução binária (binary translation)

- Neste modelo, criado pela VMware, ao invés de emular o processador, a máquina virtual é executada diretamente na CPU.
- Quando instruções privilegiadas são detectadas, a CPU irá emitir um *trap* que será tratado pelo hypervisor. No entanto, há uma série de instruções x86 que não podem ser tratadas com *traps*, por exemplo: pushf / popf.
- Para lidar com estes casos, uma técnica chamada de tradução binária foi desenvolvida. Neste modelo, o hypervisor analisa a memória da máquina virtual e intercepta essas chamadas antes de serem executadas e dinamicamente reescreve o código na memória.
- O kernel do sistema operacional não tem conhecimento da mudança e funciona normalmente.Esta combinação de *trap-and-execute* e tradução binária permite que qualquer sistema operacional possa executar inalterado em cima do hypervisor.
- Essa abordagem é complexa de implementar, porém rendeu significativos ganhos de desempenho em relação a emular totalmente uma CPU.

O Xen chegou chegando

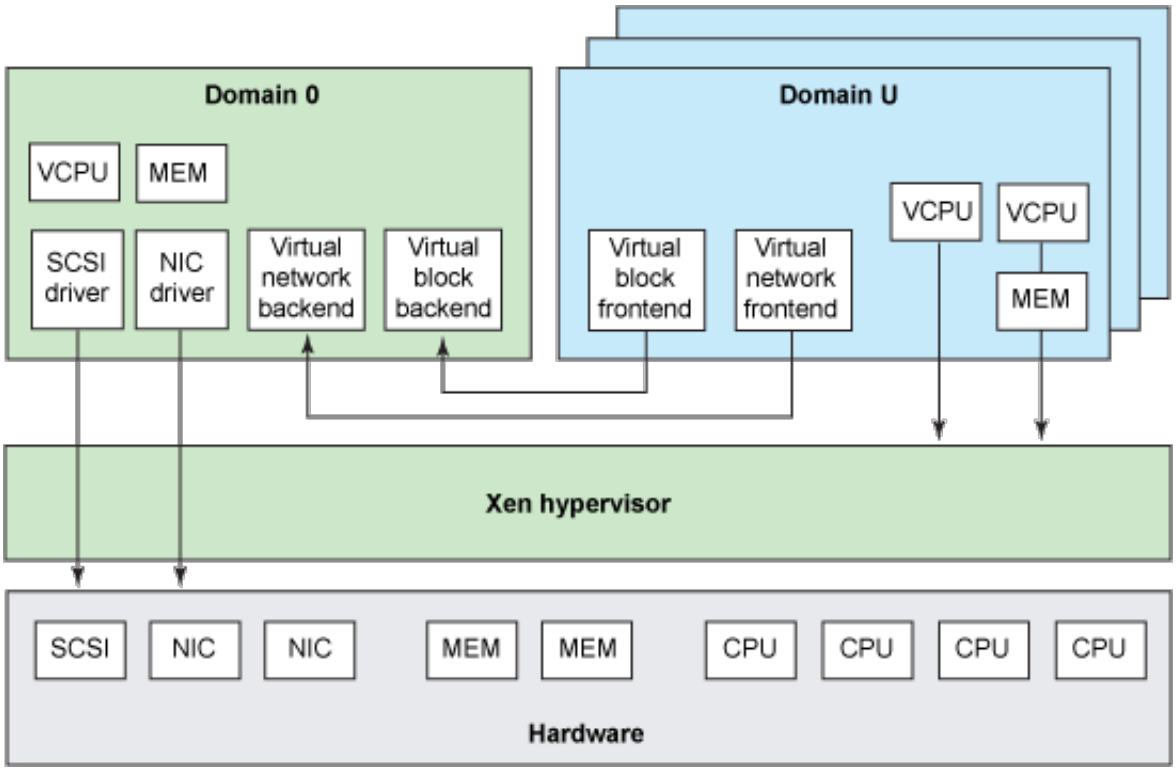
- 2002/2003: Xen 1.0 se torna público com o paper Xen and the Art of Virtualization, cria o termo paravirtualização.
    - Usando o Linux 2.4 para demonstrar a técnica, bateu e muito no VMware em performance.
- 2004:
    - Debian inclui Xen 1.2 na unstable
    - Pesquisadores de Cambridge fundam a XenSource, para que tornar o Xen um

produto corporativo e não apenas o resultado de trabalho acadêmico. Projeto continua como GPL e relativamente aberto à comunidade.
    - Xen 2.0: Linux 2.4.27/2.6.9 and NetBSD: http://lwn.net/Articles/109789/
- 2005
    - Junho: Fedora Core 4 inclui suporte a um snaphost do Xen 3.0
    - Novembro: Sun começa a portar Solaris para o Xen: http://virtualization.info/en/news/2005/11/xen-to-be-integrated-in-solaris-10.html
    - Dezembro:
        - Xen 3.0 é lançado: http://lwn.net/Articles/162841/
        - **Intel lança CPUs Xeon com VT-x**

Paravirtualização

- Em vez de lidar com uma instruções privilegiadas, a abordagem da paravirtualização é de modificar o sistema operacional em execução na máquina virtual e substituir todas as instruções privilegiadas com chamadas diretas ao hypervisor.
- Neste modelo, o sistema operacional modificado está ciente de que ele está sendo executado em um hypervisor e pode cooperar com o hypervisor para agendamento de I/O, eliminando a necessidade de emular dispositivos de hardware como placas de rede e controladores de disco.

Xen



Xen

- A plataforma Xen é composta por dois componentes: o hypervisor Xen, que é responsável pelo núcleo de atividades como CPU, virtualização de memória, gerenciamento de energia e escalonamento de máquinas virtuais.
- O hypervisor Xen carrega uma máquina virtual privilegiada especial chamada Domínio0 ou dom0. Esta máquina virtual tem acesso direto ao hardware e fornece drivers de dispositivo e gestão de I/O para as máquinas virtuais.
- Cada máquina virtual, conhecida como um domínio sem privilégios ou domU, contém uma versão modificada do kernel do Linux que ao invés de comunicar-se diretamente

com interfaces de hardware, fala com o hypervisor Xen.

- CPU e acesso à memória são tratados diretamente pelo hypervisor Xen, mas I/O é direcionado para o domínio 0. O kernel do Linux inclui o "front end" dispositivos de rede e bloco I/O. Os pedidos de I/O são passados para o "back-end" no domínio do processo 0, que gerencia o I/O.
- Neste modelo, o kernel hóspede, em domU é executado em ring 1, enquanto o espaço de usuário executado no anel 3.
- Enquanto o Xen é geralmente categorizada como sendo uma solução *Type 1* de hypervisor, toda a plataforma exige um sistema operacional Dom0 para acessar o hardware.

Surge o KVM

- 2006
  - Março: Fedora Core 5 com Xen 3.0. SUporte HVM precisa de CPU com virtualização.
  - Maio: AMD lança CPUs com suporte a virtualização.
  - Julho
    - VMware vai suportar Xen: http://virtualization.info/en/news/2006/07/vmware-could-decide-to-support.html
    - Novell lança SuSE Enterprise 10 com Xen, primeira empresa fora XenSource a dar suporte corporativo ao Xen.
  - Outubro: KVM é submetido para o Linux pela Quamranet: https://lkml.org/lkml/2006/10/19/146

Descrição do KVM

```
The following patchset adds a driver for Intel's hardware
virtualization
extensions to the x86 architecture.  The driver adds a character device
(/dev/kvm) that exposes the virtualization capabilities to userspace.
Using
this driver, a process can run a virtual machine (a "guest") in a fully
virtualized PC containing its own virtual hard disks, network adapters,
and
display.

Using this driver, one can start multiple virtual machines on a host.
Each
virtual machine is a process on the host; a virtual cpu is a thread in
that
process.  kill(1), nice(1), top(1) work as expected.
```

KVM em funcionamento

```
#include <linux/kvm.h>

open("/dev/kvm")
ioctl(KVM_CREATE_VM)
ioctl(KVM_CREATE_VCPU)
for (;;) {
     ioctl(KVM_RUN)
     switch (exit_reason) {
     case KVM_EXIT_IO:  /* ... */
```

```
    case KVM_EXIT_HLT: /* ... */
    }
}
```

- Mais detalhes em: http://blog.vmsplice.net/2011/03/qemu-internals-big-picture-overview.html

KVM

```
modprobe kvm
modprobe kvm_intel ou kvm_amd
qemu-img create -f qcow vm-disk.img 4G
kvm -m 384 -cdrom guestos.iso -hda vm-disk.img -boot d
```

KVM

- KVM é implementado como um módulo de kernel carregável que converte o Linux em um hypervisor *bare-metal*.
- Há dois princípios fundamentais de projeto que ajudaram o KVM a amadurecer rapidamente em um hypervisor estável e de alto desempenho.

Use o hardware, Luke

- O KVM foi projetado após o advento da virtualização assistida por hardware.
- O hypervisor KVM requer processadores com Intel VT-x ou AMD-V habilitados e utiliza esses recursos para virtualizar o CPU. Ao exigir o suporte de hardware ao invés de apenas otimizar com o mesmo se disponível, o KVM foi capaz de projetar uma solução otimizada sem a necessidade de modificações no sistema operacional virtualizado.

Não reinvente a roda

- Há muitos componentes que um hypervisor exige, além da capacidade de virtualizar a CPU e memória, por exemplo
  - Gerenciador de memória
  - Escalonador de processos
  - Pilha de I/O e rede
  - **Drivers de dispositivo**
  - Gerenciamento de segurança.

Simplificando

- Na verdade um hypervisor é realmente um sistema operacional especializado, diferindo apenas dos de uso geral pois que se executam máquinas virtuais ao invés de aplicações.

Por que o Linux?

- Uma vez que o kernel do Linux já inclui os principais recursos exigidos por um hypervisor e foi amadurecido em uma plataforma da madura e estável por mais de 15 anos de apoio e desenvolvimento, é mais eficiente de construir sobre essa base ao invés de escrever todos os componentes necessários, tais como um gerenciador de memória,

escalonador, etc a partir do zero.

- Neste contexto, o KVM se beneficiou da experiência do Xen. Um dos principais desafios do arquitetura do Xen é a arquitetura da divisão de domínio0 e o hypervisor Xen. Desde o hypervisor Xen fornece os recursos da plataforma central dentro da pilha, ele tem a necessidade de implementar esses recursos, como escalonador e gerenciador de memória a partir do zero.

Por que o Linux?

- Por exemplo, enquanto o kernel do Linux possui um gerenciador de memória madura e comprovada, incluindo suporte para NUMA e sistemas de grande escala, o hypervisor Xen necessita construir este apoio a partir do zero. Da mesma forma recursos como gerenciamento de energia que já estão maduros e comprovado em campo no Linux tinha que ser

re-implementado no hypervisor Xen.

- Outra decisão importante tomada pela equipe KVM era incorporar o KVM no kernel do Linux o quanto antes. O código KVM foi apresentado à comunidade do kernel Linux em dezembro de 2006 e foi aceito em o kernel 2.6.20, em janeiro de 2007.
- Neste ponto KVM tornou-se parte do núcleo do Linux e é capaz de herdar recursos chave do kernel do Linux.

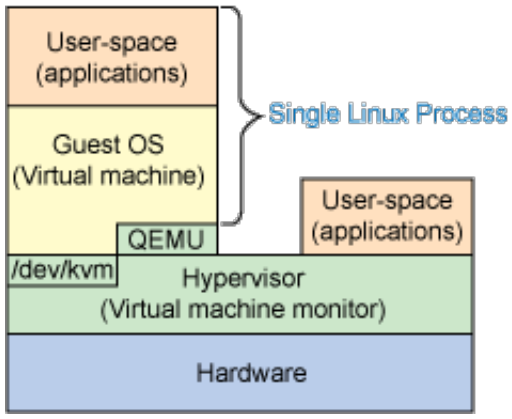
Por que o Linux?

- Em contrapartida, as correções necessárias para construir o Domínio0 Linux para o Xen ainda não faz parte do kernel Linux e obriga os fornecedores a criar e manter um

fork do kernel Linux.

- Isto tem levar a um aumento dos encargos com os distribuidores de Xen, que não podem alavancar facilmente os recursos do montante kernel. Qualquer novo recurso de correção de bug ou um patch para o kernel acrescentado a montante deve ser back-portados para trabalhar com o patch define Xen.

Arquitetura KVM



Arquitetura KVM

- I/O e dipositivos são virtualizados através de um processo QEMU levemente modificado (cara processo executa uma máquina virtual).
- Execução de I/O de um sistema operacional convidado é fornecido com o QEMU.
- QEMU é uma solução de virtualização de que permite a virtualização de um ambiente de PC inteiro (inclusive discos, placas gráficas, dispositivos de rede). Quaisquer solicitações de I / O de um sistema operacional convidado faz são interceptadas e direcionadas para o modo de usuário para ser emulado pelo processo QEMU.

Arquitetura KVM

- KVM permite a virtualização de memória através do dispositivo **/dev/kvm**.
- Cada sistema operacional convidado tem seu próprio espaço de endereço que é mapeado quando o convidado é instanciado.
- A memória física que é mapeada para o sistema operacional guest é a memória virtual mapeada para o processo.

KVM e Xen iniciam competição

- 2007
  - Fevereiro:

        - Linux 2.6.20 é lançado com KVM incluído (Xen não existia upstream).
        - KVM começa a fazer barulho: KVM steals virtualization spotlight
    - Março: Red Hat Enterprise 5 lançado com suporte a Xen 3.0.
    - Abril: Entrevista com Avi Kivity: http://kerneltrap.org/node/8088
    - Julho: Microsoft anuncia parceria com a XenSource na especificação de um para Hypervisor (o que viria a ser o Hyper-V) e HP apoia: http://virtualization.info/en/news/2006/07/microsoft-to-support-xen-virtual.html
    - **Agosto: Citrix compra XenSource por US$ 500 milhões (produtos como XenServer entre outros).**
    - Novembro: Oracle VM, baseado no Xen: http://virtualization.info/en/news/2007/11/oracle-announces-its-own-xen-based.html

KVM e Xen iniciam competição

- 2008
    - Abril: Canonical adota KVM como sua solução de virtualização: http://www.theregister.co.uk/2008/02/11/ubuntu_hardy_heron_kvm/
    - Junho: Microsoft lança Hyper-V, "compatível" com Xen: http://virtualization.info/en/news/2008/06/microsoft-hyper-v-day-after.html
    - Agosto: Lançado Xen 3.3.
    - Setembro: Red Hat compra Quamranet: http://www.redhat.com/about/news/prarchive/2008/qumranet.html
- 2009
    - Setembro: Red Hat incluí KVM no RHEL 5.4 junto com Xen: http://arstechnica.com/open-source/news/2009/09/red-hat-moves-forward-with-kvm-virtualization-in-rhel-54.ars
- 2010
    - Março: Novell vai suportar KVM no SLES 11 e diz que KVM é o futuro: http://www.h-online.com/open/news/item/Novell-supports-KVM-963031.html
    - Abril: RHEL 6 não terá Xen como hypervisor: http://www.computerworld.com/s/article/9175883/Red_Hat_drops_Xen_from_RHEL

Porque KVM

- Linux é o Host/Hypervisor/VMM
- Máquinas virtuais são simples processos.
- kill, top e outros comandos podem ser utilizados normalmente.
- Utilização dos escalonadores de I/O, memória e CPU do Linux.

- Usa modelo de segurança do Linux
  - As máquinas virtuais rodam com privilégios de usuário.
- Drivers paravirtualizados para melhor performance onde necessário.
- Possibilidade de fazer swap com a memória da máquina virtual.
- NUMA
- Se beneficia diretamente de todas as melhorias que o Linux recebe.

Paravirtualização não faz mais sentido?

```
From: Alok Kataria <akataria-AT-vmware.com>
To: Ingo Molnar <mingo-AT-elte.hu>, Thomas Gleixner <tglx-AT-
linutronix.de>, "H. Peter Anvin" <hpa-AT-zytor.com>

We ran a few experiments to compare performance of VMware's
paravirtualization technique (VMI) and hardware MMU technologies
(HWMMU)
on VMware's hypervisor.
...
In most of the benchmarks, EPT/NPT (hwmmu) technologies are at par or
provide better performance compared to VMI.
The experiments included comparing performance across various micro and
real world like benchmarks.
...
In light of these results and availability of such hardware, we have
decided to stop supporting VMI in our future products.
```

- http://lwn.net/Articles/353853/

paravirt_ops

- Mais coisas sobre isso nos comentários: http://lwn.net/Articles/353852/
- Comentários muito bons sobre Xen VS KVM: http://lwn.net/Articles/374191/

Resumindo

- O que é KVM: Linux + drivers para CPUs Intel e AMD + QEMU
  - Simples de usar, rápido e robusto.

Ferramentas de gerenciamento

- libvirt (Virt-Manager, virsh)
- Ganeti

- OpenNebula
- Eucalyptus
- OpenStack
- OpenQRM
- ConVirt
- Red Hat Enterprise Virtualization
- Proxmox
- Enomaly
- OpenNode
- oVirt
- Karesansui
- Univention Virtual Machine Manager

Referências

- Paper sobre VT-x: http://download.intel.com/technology/itj/2006/v10i3/v10-i3-art01.pdf
- Discover the Linux Kernel Virtual Machine: http://www.ibm.com/developerworks/linux/library/l-linux-kvm/
- Documento interessante sobre KVM, acho que dá um bom overview: http://www.redhat.com/f/pdf/rhev/DOC-KVM.pdf
- http://blog.vmsplice.net/2011/03/qemu-internals-overall-architecture-and.html
- Slides sobre KVM: http://www.geeksofpune.in/drupal/files/8025301-kvmway.pdf
- http://www.linux-kvm.org/wiki/images/4/42/Kvm-device-assignment.pdf
- kvm: the Linux Virtual Machine Monitor: http://www.kernel.org/doc/ols/2007/ols2007v1-pages-225-230.pdf