关键名词：

完全虚拟化，敏感指令在操作系统和硬件之间被捕捉处理，客户操作系统无需修改，所有软件都能在虚拟机中运行，例如IBM CP/CMS，VirtualBox，VMware Workstation

硬件辅助虚拟化，利用硬件（主要是CPU）辅助处理敏感指令以实现完全虚拟化的功能，客户操作系统无需修改，例如VMware Workstation，Xen，KVM

部分虚拟化，针对部分应用程序进行虚拟，而不是整个操作系统

准虚拟化/超虚拟化（paravirtualization），为应用程序提供与底层硬件相似但不相同的软件接口，客户操作系统需要进行修改，例如早期的Xen

操作系统级虚拟化，使操作系统内核支持多用户空间实体，例如Parallels Virtuozzo Containers，Unix-like系统上的chroot，Solaris上的Zone

KVM的虚拟化技术和原来的XEN虚拟化技术有什么区别，而且在今后redhat是否不会在redhat里在集成XEN的功能，而转向对KVM的支持。

答：XEN目前支持Full Virtualization(全虚拟化) 和 Para Virtualization.

Full Virtualization的好处在于现有的x86架构的操作系统可以不用修改，直接运行在虚拟机上。 Para Virtualization的好处是性能好，但是虚拟机上运行的操作系统内核要修改。

目前主流的厂家的虚拟化重点都是放在Full Virtualization上面。

KVM采用的是Full Virtualization，需要CPU支持VT。 如何确认你的CPU是否支持VT,查看cpu flag, intel cpu flag 会有 "vmx" , amd cpu flag 会有 "svm".

从架构上讲，xen是自定制的hypervisor，对硬件的资源管理和调度，对虚拟机的生命周期管理等，都是从头开始写的。 KVM全称是Kernel-based Virtual Machine, kernel代表的是Linux kernel。KVM是一个特殊的模块，Linux kernel加载此模块后，可以将Linux kernel 变成hypervisor，因为Linux kernel已经可以很好的实现对硬件资源的调度和管理，KVM只是实现了对虚拟机生命周期管理的相关工作。 KVM的初始版本只有4万行代码，相对于xen的几百万行代码显得非常简洁。

更多关于KVM架构的信息，请参考KVM白皮书:

http://www.linuxinsight.com/files/kvm\_whitepaper.pdf

红帽在2007年发布RHEL5，采用的是xen来提供虚拟化功能。从红帽RHEL5.4开始，xen和kvm同时存在。 RHEL5上的xen，红帽会支持到2014年。 后续红帽的重点会放在KVM上面。

linux上两种免费的开源虚拟化管理技术：Xen 和KVM

XEN出现较早，Xen是“第一类”运行在裸机上的虚拟化管理程序（Hypervisor），也是当前相当一部分商业化运作公司的基础技术，其中包括Citrix系统公司的XenServer和Oracle的虚拟机。Xen技术的倡导者们声称Xen的性能强劲，并且拥有一个广泛的管理工具和能够交付卓越性能，以及其它诸多优点。

但是最近有一些Linux厂商——包括Red Hat和Canonical把基于内核的虚拟机（KVM：Kernel-based Virtual Machine）技术内置在RHEL和Ubuntu中。KVM是一个轻量级的虚拟化管理程序模块，该模块主要来自于Linux内核。虽然只是后来者，但是由于其性能和实施的简易性，以及对Linux重量级的持续支持，所以还是有必要认真对待KVM技术的。

最开始时使用的vmware，98时代的vmware3

vmware只是一个模拟器

full virtualization 完全虚拟化 模拟器

para virtualization

还有一种不能算是虚拟化的container 容器

完全虚拟化，

在一个操作系统的基础上再安装一个软件，通过模拟的方式运行在自己模拟的基础上。转换翻译为对应的指令让宿主机执行，如果需要执行一些特权指令，那么就需要

每一条指令都需要转换所以执行效率比较低，性能差的多硬件平台和实际平台可以不一致，X86的你可以模拟为Power PC，只不过是执行效率比较低罢了。CPU是可以模拟的。

准虚拟化 半虚拟化：

在裸机上直接安装了一hypervisor,这个虚拟化软件来完成硬件资源的模拟，然后输出为各个操作系统，操作系统相当于在hypervisor上的一个软件，hypervisor 非特权指令直接在cpu上执行，不需要转换，虚拟机之间不互相影响。只转换特权指令。特权指令大概只有10几个

至于那个好：半虚拟化

但是半虚拟化运行的平台，例如申请内存就不是直接和内存打交道，需要向hypervisor申请，，这样操作系统必须知道自己不是直接操作硬件而是和hypervisor打交道，这个操作系统必须做出修改，让他明白他是运行在hypervisor上。必须对应用在这个之上的操作系统内核做出修改。所以是不是所有的系统都支持

这种虚拟化技术让性能提升很多，接近于直接操作物理机，

X84平台上CPU指令级别分为四层，，但是linux只使用了2层，普通层和特权层，内核本身是内核层0123层第三层是用户空间，

这种虚拟化上Hypervisor运行在0层，内核只能在hypersivor基础上运行，

硬件虚拟化：需要修改cpu来实现不修改操作系统实现半虚拟化。在4层的基础上增加了第-1层。以后hypersivor运行在-1层，这样内核还是第0层，不需要修改，这样的技术就叫做硬件虚拟机。HVM

只有硬件支持的话才可以实现基于hypervisor的硬件虚拟化

Hypervisor是直接安装在硬件上的，需要解决硬件驱动，但这样不就成了操作系统了么？就必须提供各种硬件驱动就成了操作系统了，这样提及比较大了，这样似乎看起来和在普通操作系统上运行就差不多了。但是他们尽量的做小，XEN可能才4M的空间。

他们将每一个运行在之上的操作系统都作为一个domain,dom0,dom1dom2…..

Hypervisor自身不提供任何硬件驱动

Dom0是第一个运行在操作系统的软件，硬件驱动都在dom0上，dom有自己的内核，用户空间，它是由特权的可以再创建dom123.。。。dom1想查找网卡，hypervisor去dom0上找驱动，然后提供驱动，dom0提供驱动，也提供了管理其他机器的借口，XEN是半虚拟化，但是也可以做完全虚拟化。XEN只能运行在linux上，而且XEN的安装比较奇怪，先安装linux再安装xen内核，然后之后的操作系统再运行在xen之上

Xen流行的状况在kvm出现后发生了改变

KVM kernel-based virtual machine

XEN和ESX之类的是需要先安装hypervisor，在提供的安装接口上然后再安装操作系统。

Xen有点特别就是需要借助linux先安装

Kvm是linux的一个内核模块，如果内核有这个功能的话就可以直接将内核当做hypervisor来用，2.6.3之后直接整合到内核中了，这样一个虚拟机就像一个进程一样，可以直接kill之类。内有kvm之前，内存分为两种模式，用户空间，内核空间。之后就分为：guest。。。 这个虚拟机就只是一个进程

要想使用kvm1第一必须要支持硬件虚拟化，本身就是基于linux内核的，优化也是不少的。

（但网上有人反映xen要好很多）。Kvm这个公司被红帽收购，后来直接做进内核，

总结一下：完全虚拟化

Kvm **完全虚拟化需要HVM硬件虚拟化**

Guest不加修改就能运行就叫做完全虚拟化

Vmware workstation 完全虚拟化，也可以看做一个模拟器

类似的模拟器 qemu等

X86上模拟非X86

XEN也是完全虚拟化

Vshpere也是完全虚拟化（HVM）

XEN也支持半虚拟化

所谓的虚拟化就是在一个硬件上先安装一层管理软件，这个管理软件可以运行虚拟机，将我们硬件虚拟为几个可以独立运行的实例。

下一个问题

虚拟化还有一种方式：container

Chroot实质就是一个容器！

虚拟用户空间

性能：

Container肯定最好，因为这就是在本地运行，container在linux有名的软件就是OpenVZ

不稳定，任何一个搞崩溃了内核都会直接崩溃

硬件还是硬件 软件 还是软件 内核还是内核，不过在linux装了windows的库文件，就可以使用例如wine，模拟了用户空间 cywin可以在windows运行linux

###################################################

kvm 07年诞生，08年被红帽收购 2.6.20之后被做进内核

完全虚拟化，不需要改动os内核，不过完全虚拟化的能努力会稍弱，但是现在的VT扩展之后性能也越来越好，

xen 03年最开始时半虚拟化，现在也支持全虚拟化。不支持VT技术的cpu只能使用半虚拟化。半虚拟化需要hypervisor配合，性能稍微强一些。os开发时包括进去，os使用cpu内存的时候直接使用，不需要仿真。但是xen有自己的进程调度器，存储管理模块，代码庞大。

Vmware 的ESX系列也是半虚拟化。

国内的盛大云、阿里云、万网云使用的都是xen

XEN的配置工具：virt-manager vort-install xm

分别为管理 安装 启动

qemu-kvm libvirt virt-install

kvm的具体配置过程

########################################################

hyperv和esx系列都是基于硬件支持的

hyperv采用微内核结构，esx采用的是单内核结构

单内核所有的硬件驱动都集中在hypervisor层，被所有的虚拟机共用，够提供良好的性能但是安全性和兼容性有缺陷，而且体积庞大，微内核里面没有驱动

########################################################

根据前人的经验：xen kvm vsphere 还是vsphere最成熟bug最少，就是有点贵

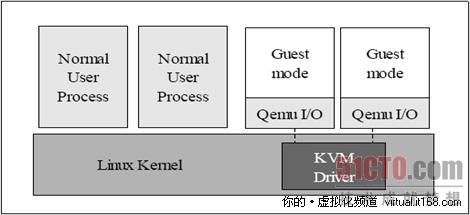
xen和kvm适合专业团队做二次开发，自由度高bug不少。出现问题需要专业团队解决，vsphere方案成熟，功能强大适合愿意花钱的中小型企业

##############################################

10年的RHEL6beta中去掉了XEN，

在KVM模型中，每一个虚拟机都是一个由linux调度程序管理的标准进程，

KVM使用的方法是通过简单的加载内核模块而**将linux内核转换为一个系统管理程序**，这个内核模块导出了一个名为/dev/kvm的设备，有了这个设备VM使得自己的地址空间独立于内核或者运行着其他VM的地址空间。设备树中的所有设备对所有用户空间进程来说是通用的。但是每个打开/dev/kvm的进程看到的是不同的映射。安装 KVM 之后，您可以在用户空间启动客户操作系统。每**个客户操作系统都是主机操作系统(或系统管理程序)的一个单个进程。**



KVM和QEMU的关系：

准确来说KVM仅仅是内核的一个模块。管理和创建完整的KVM虚拟机需要一些辅助工具。

在linux中可以使用modprobe命令加载KVM模块，如果使用rpm包安装的话是启动是自动加载。加载后之后还得使用其他工具进一步创建虚拟机。但仅有KVM模块是远远不够的，因为用户无法控制内核模块去做事情，必须有一个用户空间的工具。

这个工具KVM开发者选择了已经成型开源的QEMU。它可以虚拟不同的CPU架构。比如在X86上模拟POWER。KVM利用QEMU的X86模块改进形成了可以控制KVM内核模块的用户空间工具QEMU-KVM。所以linux发行版中分为**内核部分的KVM模块和QEMU-KVM工具**

KVM和libvirt virt-manager：

尽管有QEMU-KVM工具可以创建和管理kvm虚拟机，但是redhat为了KVM开发了更多的辅助工具，比如libvirt libguestfs等。因为QEMU的工具效率不高，不利于使用。

libvirt提供了一套多语言的接口API。为各种虚拟化工具提供了一套方便可靠的可编程接口。不仅支持KVM还支持XEN的虚拟机。秩序使用libvirt提供的函数就可以链接到宿主机。使用同样的命令控制不同的虚拟机。libvirt还提供了一套文本的虚拟机管理命令：virsh。最终用户更渴望的图形化界面virt-manager是一套python编写的虚拟机图形管理界面，它就是使用libvirt的API实现的