北京航空航天大學
BEIHANG UNIVERSITY

云计算与大数据平台

王宝会　北航软件学院

2020年9月

充电：计算机系统的可用性

- -计算机系统的可用性定义为系统保持正常运行时间的百分比。
- —MTTF/(MTTF+MTTR)100%
- —MTTF (Mean Time To Failure， 修复前平均时间)
- —MTTR (Mean Time To Restoration,平均恢复前时间)。

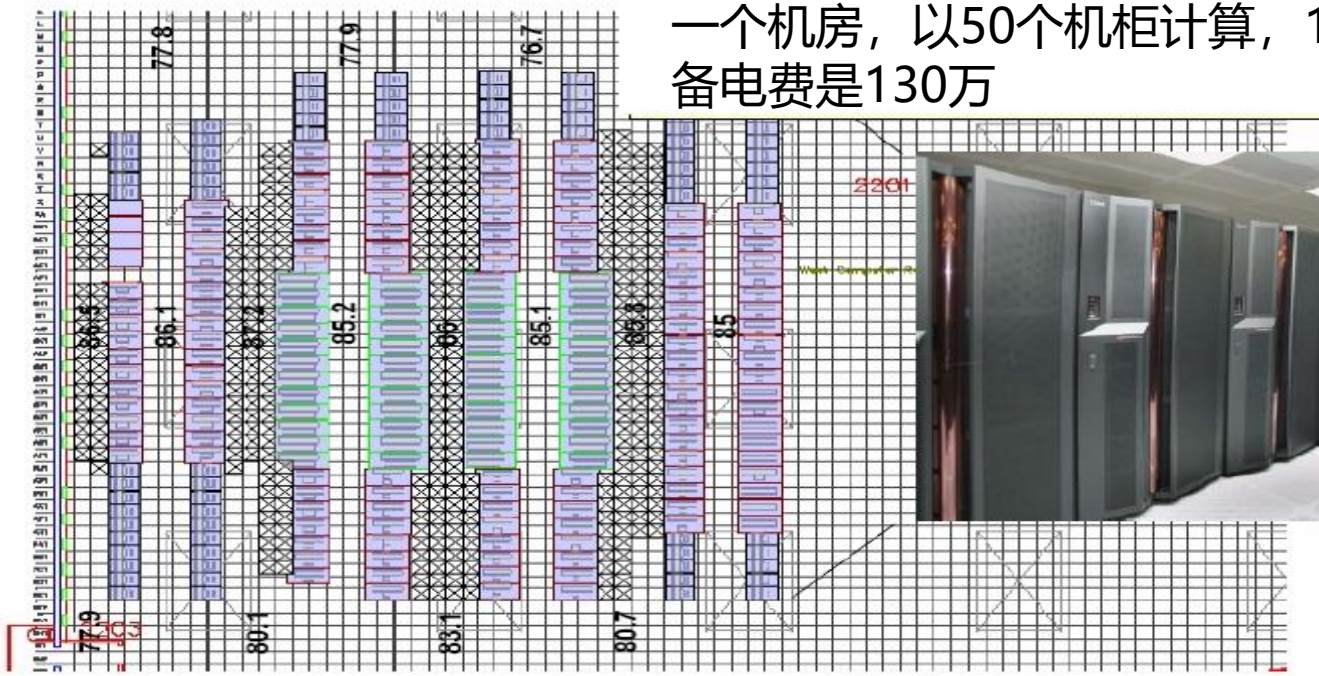
计算机产业界通常用“9”的个数来划分计算机系统可用性的类型

可用性分类	可用水平	每年停机时间
容错可用性	99．9999	< 1 min
极高可用性	99．999	5 min
具有故障自动恢复能力的可用性	99．99	53 min
高可用性	99．9	8.8 h
商品可用性	99	43.8h

充电：能源的效率对总体拥有成本起相当明显的影响

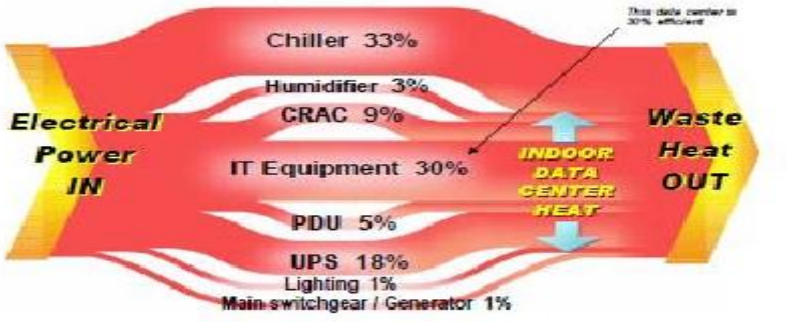
一个机柜以3KW用电量，每度电一元人民币计算，每月的电费是2160元。每年该机柜电费是2.6万。

一个机房，以50个机柜计算，150KW,一年的IT设备电费是130万



充电：数据中心的能源效率系数PUE

数据中心的用电分配



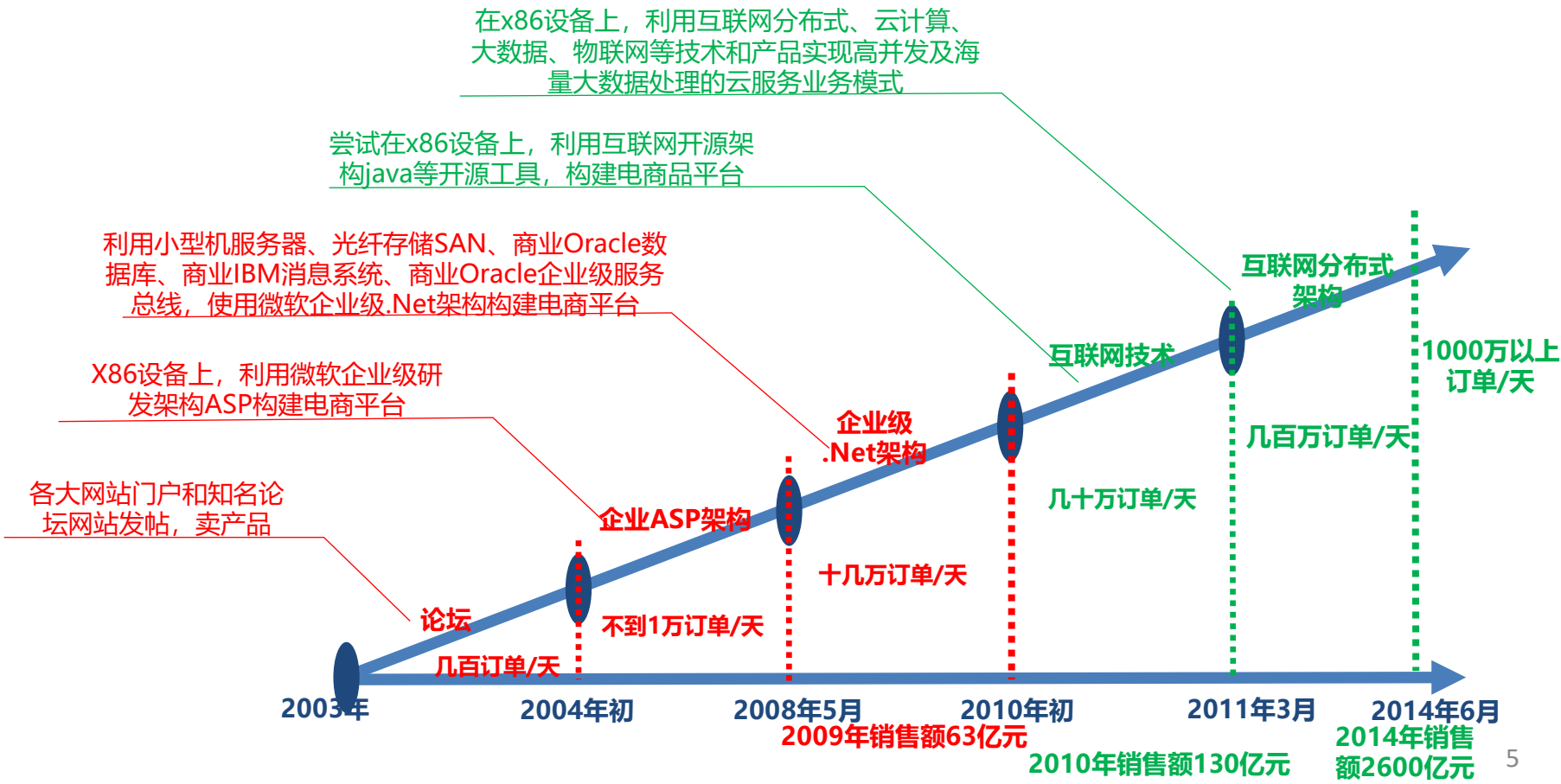
这个数据中心的能源效率为*30%*

PUE = [数据中心总用电消耗] / [IT设备能源消耗]

数据中心能源效率系数 **(PUE)**

- 理想的 ***PUE*** ***= 1.6***
- 目标 ***PUE*** ***= 2.0***
- 常见的 ***PUE*** ***= 2.4 to 2.8*** 甚至更高

云计算发展

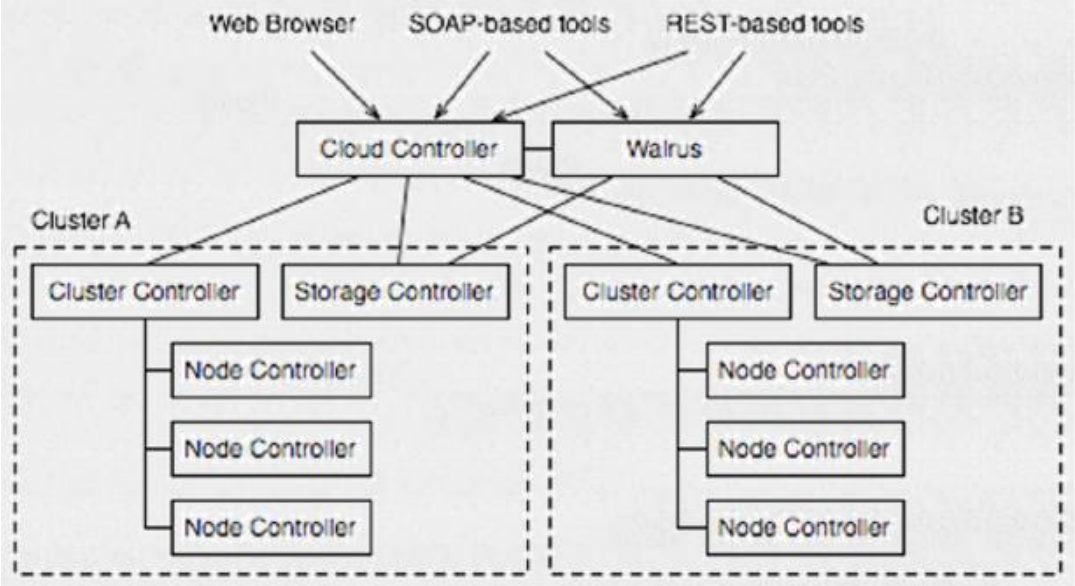

在x86设备上，利用互联网分布式、云计算、大数据、物联网等技术和产品实现高并发及海量大数据处理的云服务业务模式
尝试在x86设备上，利用互联网开源架构java等开源工具，构建电商品平台
利用小型机服务器、光纤存储SAN、商业Oracle数据库、商业IBM消息系统、商业Oracle企业级服务总线，使用微软企业级.Net架构构建电商平台
X86设备上，利用微软企业级研发架构ASP构建电商平台
各大网站门户和知名论坛网站发帖，卖产品
互联网分布式架构
互联网技术
企业级.Net架构
企业ASP架构
论坛
1000万以上订单/天
几百万订单/天
几十万订单/天
十几万订单/天
不到1万订单/天
几百订单/天
2003年
2004年初
2008年5月
2010年初
2011年3月
2014年6月
2009年销售额63亿元
2010年销售额130亿元
2014年销售额2600亿元

开源云计算系统

●Eucalyptus(桉树)

1. 是一种开源的软件基础结构，用来通
过计算集群或工作站群实现弹性的、实用的云计算(私有云)。
2.它最初是美国加利福尼亚大学Santa Barbara计算机科学学院的一个研究项目，现在已经商业化，发展成为了Eucalyptus Systems Inc。
3.与EC2和S3的接口兼容性(SOAP接口和REST接口)
4.支持运行在Xen或KVM
5.管理多集群

1.Cloud Controller(CLC)
2.Cluster Controller (cc)
3.Node Controller (NC)
4.Walrus (W)
5.Storage Controller (SC)

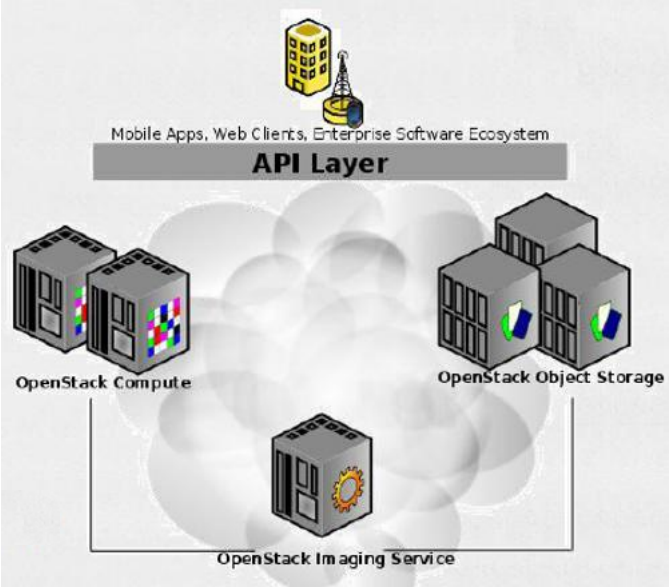


开源云计算系统

- OpenStack

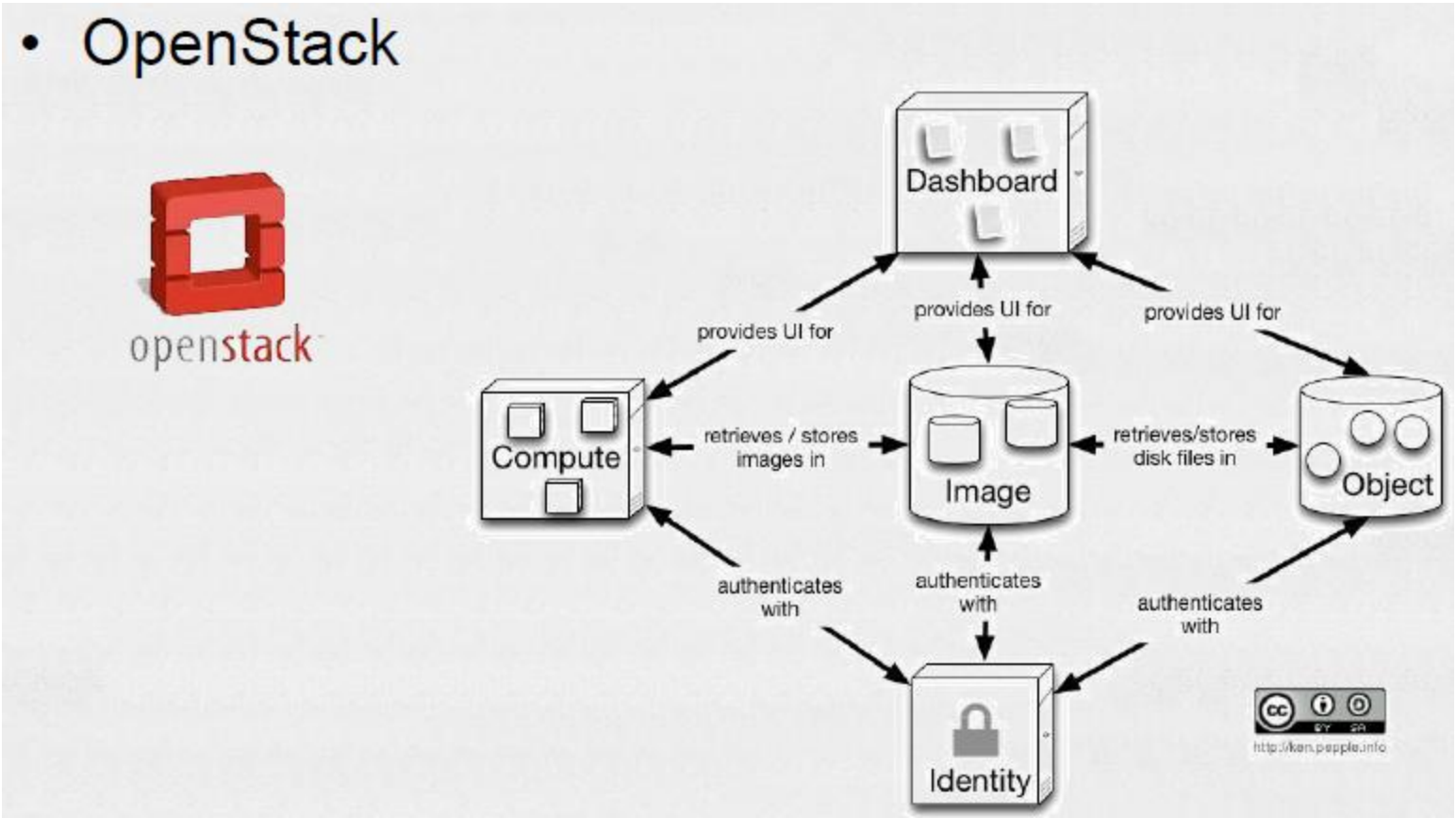
OpenStackCompute
(Nova)OpenStackObject Storage
(Swift)OpenStacklmage Service (Glance)

Dashboard
Python-nova客户端



开源云计算系统

- OpenStack

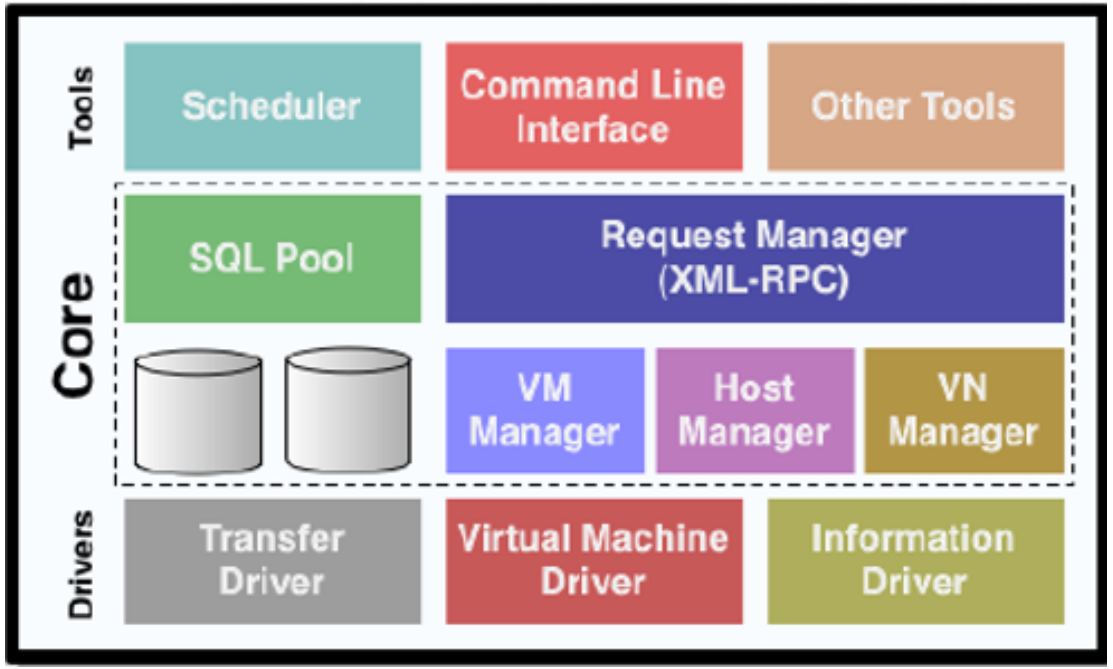


开源云计算系统

- OpenNebula

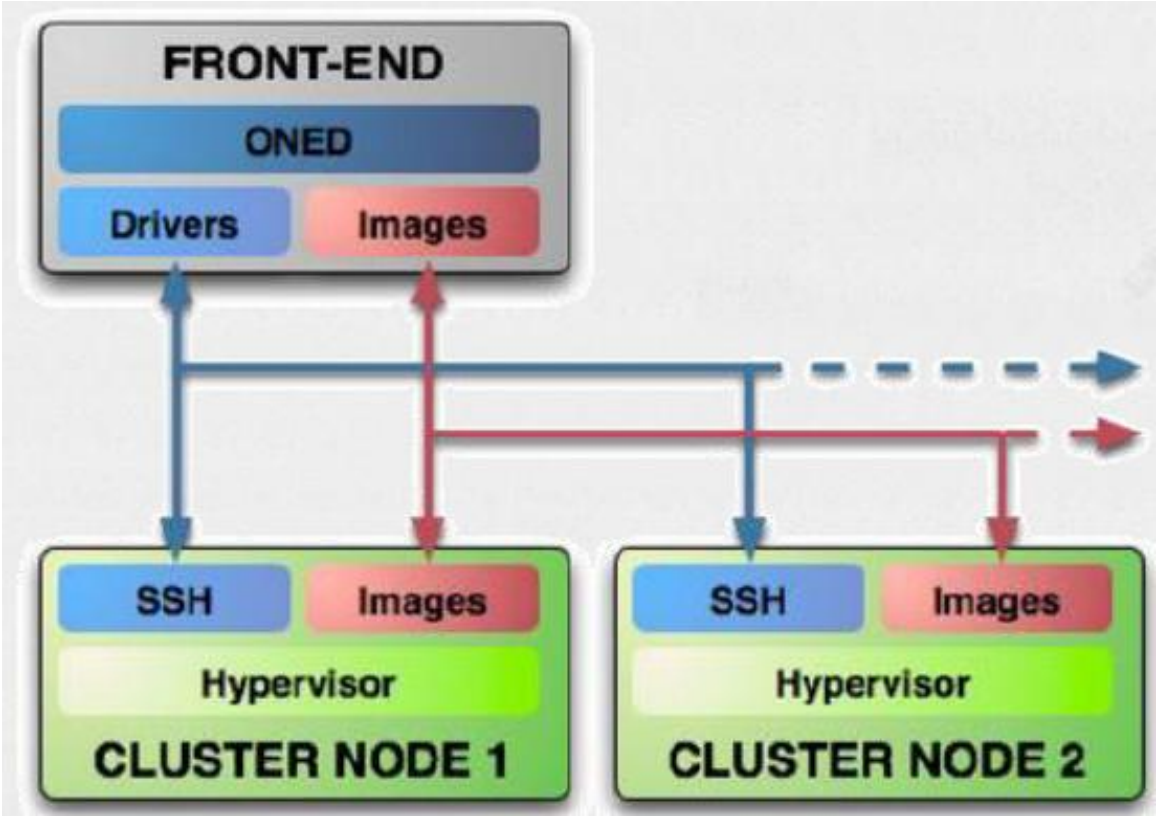
  OpenNebula.org
  The Open Source Solution for Dala Center Virlualization

可支持Xen、KVM和VMware



开源云计算系统

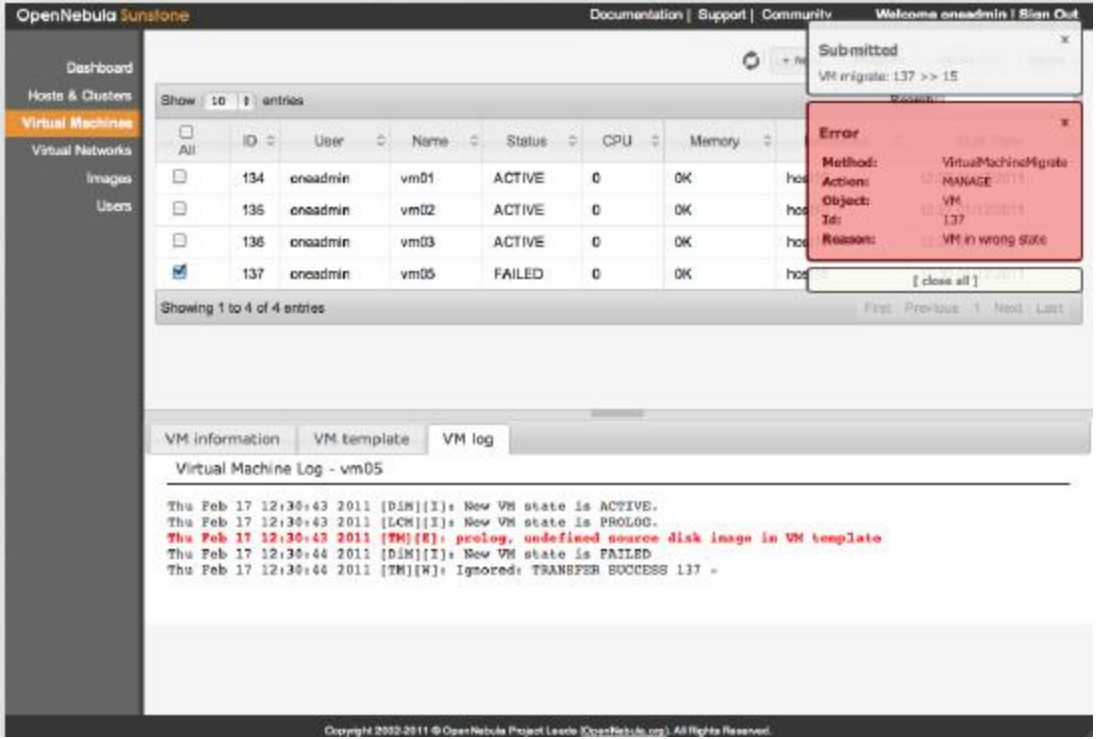
- OpenNebula



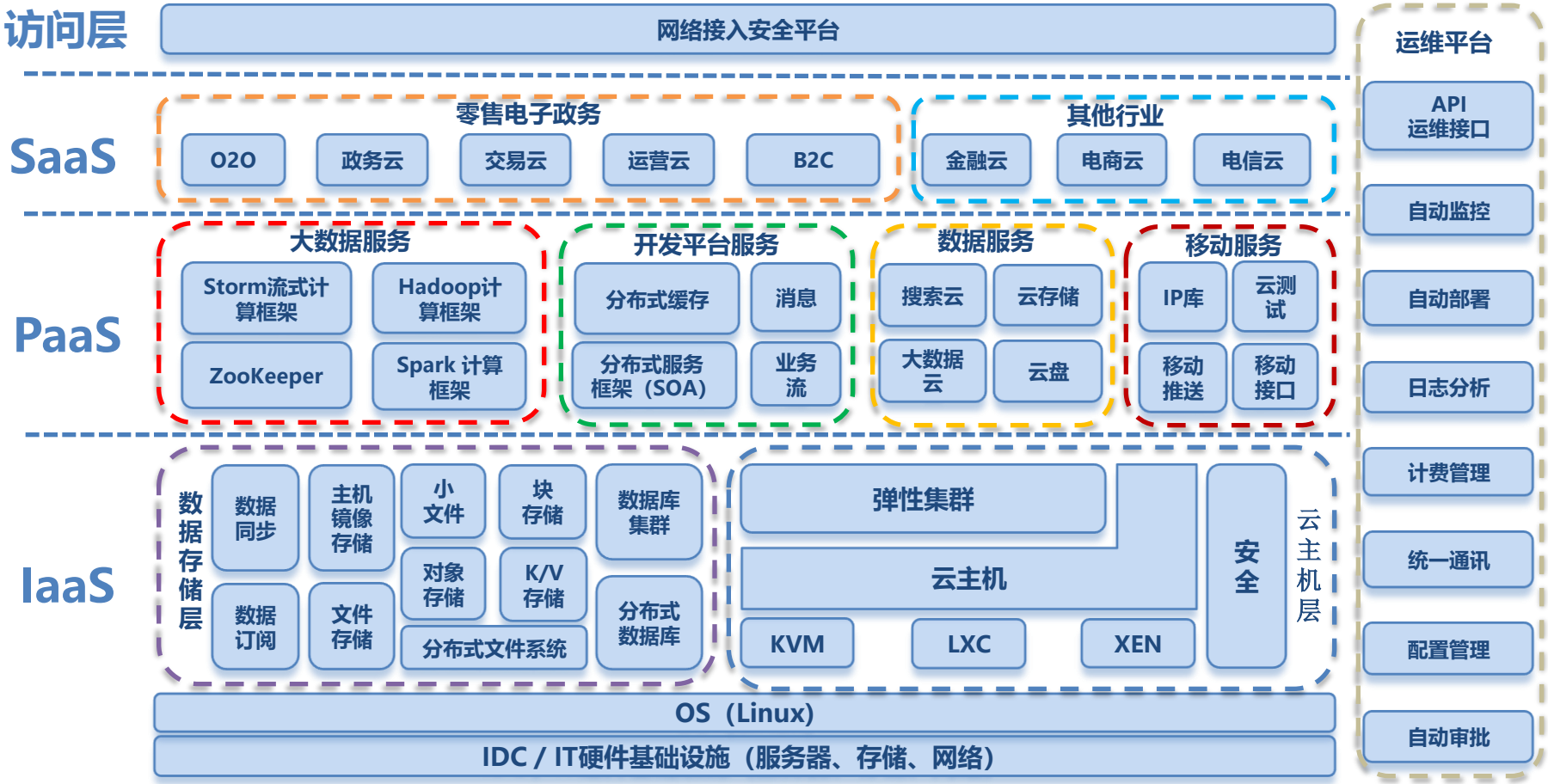
开源云计算系统

- OpenNebula

OpenNebulaSunstone



云计算架构


访问层
网络接入安全平台
SaaS
零售电子政务
O2O
政务云
交易云
运营云
B2C
其他行业
金融云
电商云
电信云
PaaS
大数据服务
Storm流式计算框架
Hadoop计算框架
ZooKeeper
Spark 计算框架
开发平台服务
分布式缓存
消息
分布式服务框架（SOA）
业务流
数据服务
搜索云
云存储
大数据云
云盘
移动服务
IP库
云测试
移动推送
移动接口
IaaS
数据存储层
数据同步
主机镜像存储
小文件
块存储
数据库集群
数据订阅
文件存储
对象存储
K/V存储
分布式文件系统
分布式数据库
弹性集群
云主机
KVM
LXC
XEN
安全
云主机层
OS（Linux）
IDC / IT硬件基础设施（服务器、存储、网络）
运维平台
API运维接口
自动监控
自动部署
日志分析
计费管理
统一通讯
配置管理
自动审批

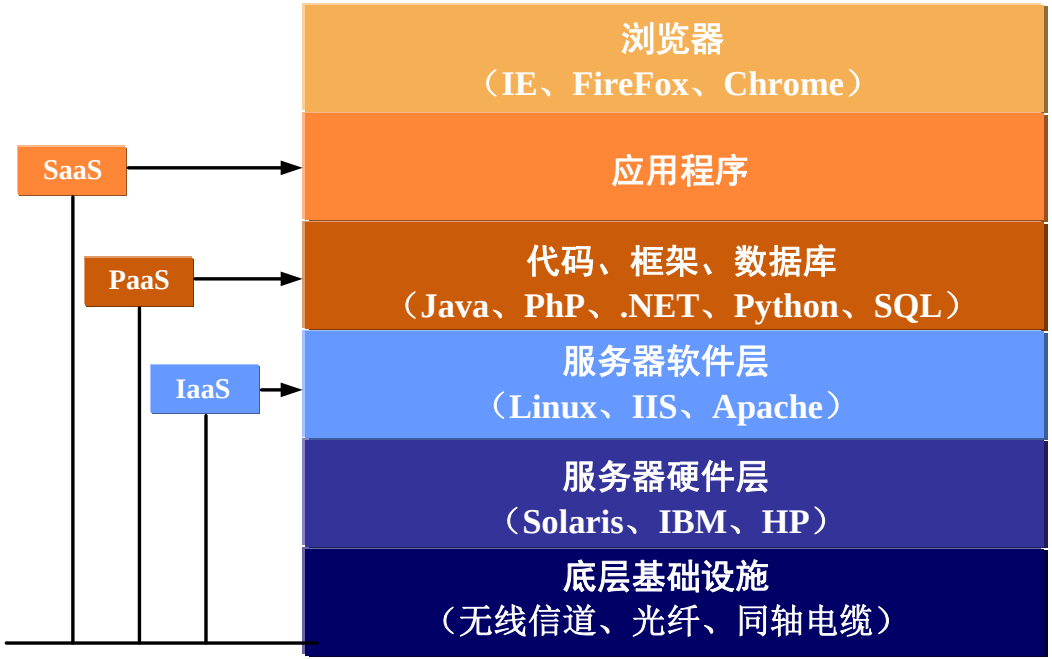
云计算带来的新的变革

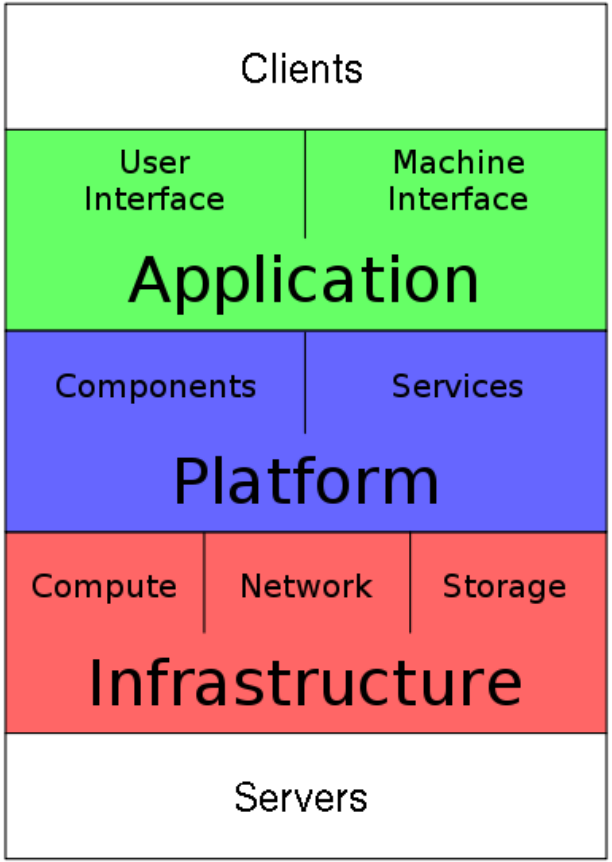
这一轮 IT 变革从终端和后端同时推进：传统的用户终端开始分化，除手机、PC 之外，开始出现电视、音箱、AR/VR 眼镜、耳机、汽车等适应不同场景、可以随时随地接入网络的多元化终端，总的趋势是终端多样化、操作系统瘦小化、浏览器中心化、网络无线化、存储处理网络化。

随着终端需要处理的数据越来越多，以及浏览器逐步成为信息交换中心，更多的存储和计算能力迁移到网上，更多的软件 Web 化，后端的服务器开始演变为“云”（大规模分布式数据中心/服务器农场）。云计算技术使后端服务器能够以较低的成本实现规模化扩展，满足海量数据的存储和并发处理需求。

技术变革使WINTEL联盟逐渐松动。

云计算的服务模式

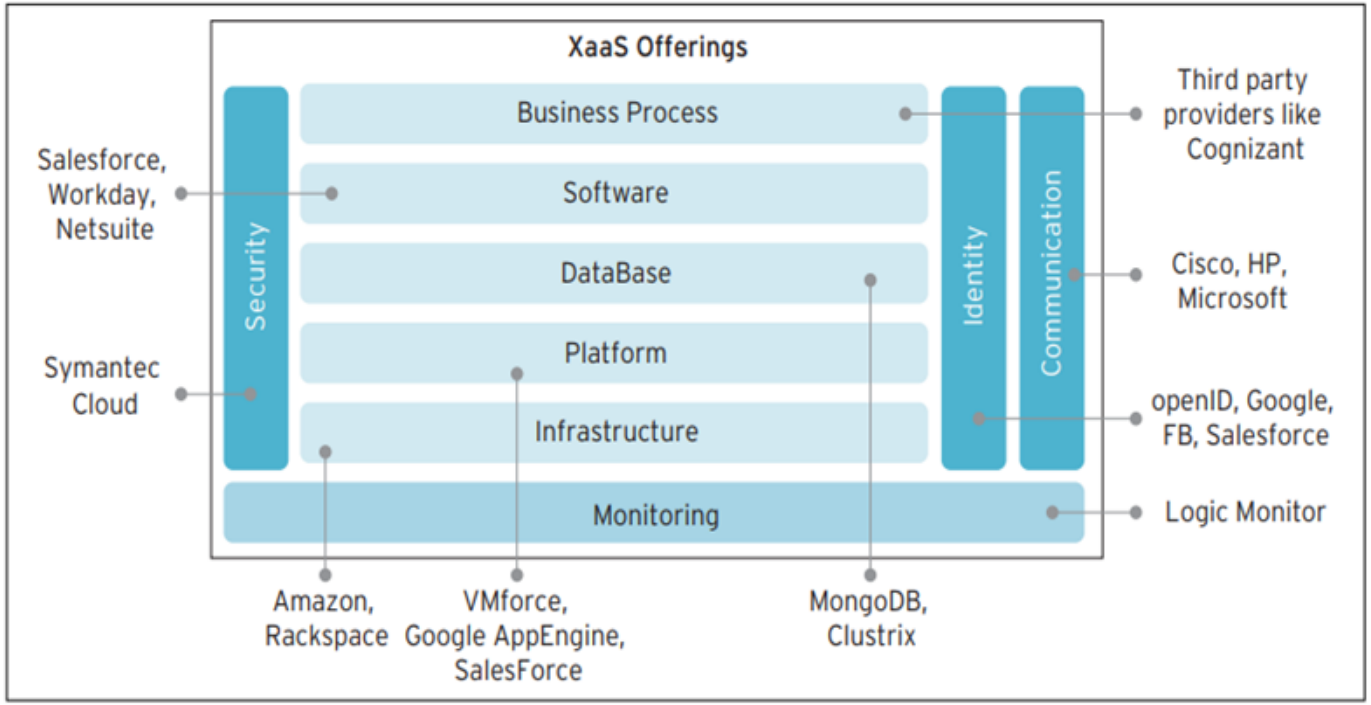




Cloud Computing Stack

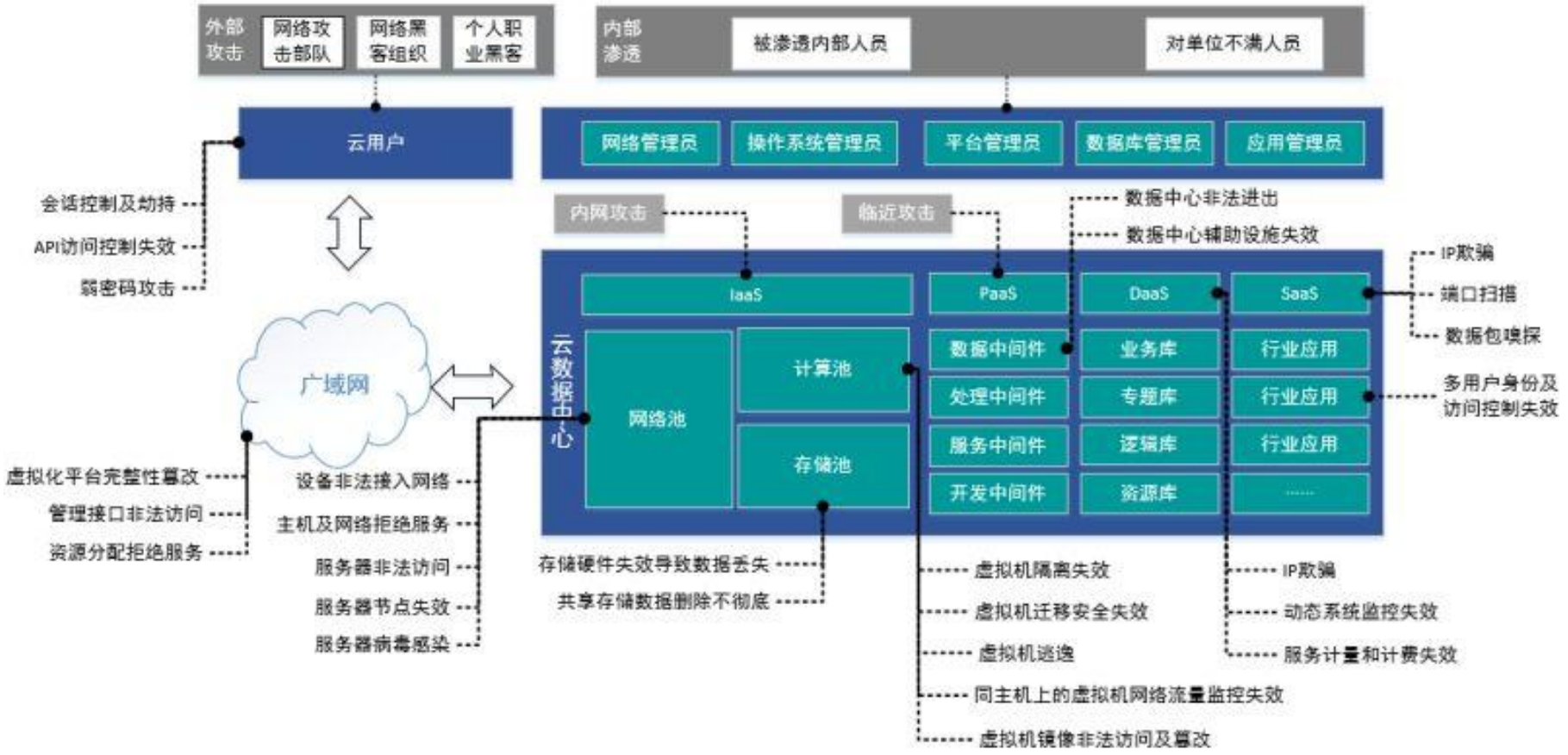
XaaS

- XaaS“一切皆服务”是一个新兴商业模型，认为任何技术设备都视为拥有提供服务的功能。



云计算安全威胁

虚拟化要求：安全设备适应云计算中心基础网络架构和应用服务的虚拟化
流量模型的转变：从分散走向高度集中，设备性能面临压力
未知威胁检测引擎的变更：客户端将从主体检测各自为战变为辅助云计算中心检测
安全边界消失



云计算身份认证问题

传统的计算模式有专门的工作人员根据资源管理者提供的用户信息进行认证，在系统可信范围进行运作，所以能更便捷的供应用户身份信息，实现身份信息的实时同步。

云域的访问服务应用广泛，在进行访问的过程中要涉及多个领域，其中不同的域都有自身特有的认证以及身份认证方式，不同域在访问控制方式上也存在差异，导致访问认证以及信息管理的统一性方面存出现分歧，造成系统的兼容性问题，对用户的访问造成困难。

云计算访问授权问题

访问授权

云计算不能应用于全部类型的访问控制模型之中。用户在进行数据存储与处理的过程中，如果将数据的整理工作外包给云服务提供商，就改变了原数据的持有者身份，失去了对数据处理所有权，

云计算通过对虚拟化技术应用实现用户之间相同硬件资源的共享，多租户技术促进用户数据集中的重要条件，通过标签把集中的数据进行隔离，进而实现相同软件资源的共享，在如此条件下造成了多租户环境现象，导致了访问控制工作要求的提高。

云计算敏感数据保护问题

问题：敏感数据的防护边界发生了变化。

云计算将一个企业的数据和应用会分布到多个数据中心，为企业创建了新的安全边界，企业需要在更多的地方实施防护，防护边界的变化增加了敏感数据安全防护的复杂度其次，资源共享机制带来了新的安全风险。

问题：数据存在泄漏风险。

云计算多租户共享资源的特点，其资源以虚拟、租用的方式供应，多个用户使用的虚拟资源很可能会绑定到同一个物理资源上。 如果云平台中的虚拟化软件中存在安全漏洞，用户的数据就存在泄露的风险。云化的数据中心网络向扁平化架构过渡， 数据中心内部之间的流量将会大大增加， 云化的数据中心和高速的Internet出口带宽可能被黑客利用作为攻击跳板，从而带来敏感数据泄露的风险

问题：数据存储模式发生了变化。

云环境下数据的存储是碎片化的，很难知道敏感数据存储的物理位置，基于物理机的数据防护手段就变得无能为力了。

云计算身份认证

通过用户名/口令的组合的身份认证技术：安全系数较低的认证方式，如果不同平台使用统一的用户名和密码，将造成用户身份信息的泄露，无法保证云环境下跨域身份认证的安全性；

通过智能卡、令牌的身份认证技术：基于 kerberos 协议的身份认证机制中，认证服务器和票据授权服务器很容易成为系统的性能和安全瓶颈，并存在密钥存储管理复杂、用户信息泄露等问题；

通过PKI技术数字证书的身份认证技术：PKI 体系是为所有网络应用提供加密、数字签名等密码服务的一种密钥管理、证书管理的体系。PKI 提供一个可扩展的基于策略的方法以实现身份鉴别和不可否认性。PKI技术两个最主要的安全技术是公钥加密技术和数字签名技术。公钥加密技术是信息的保密性和访问控制的有效手段，它为公钥加密后的数据安全性提供了保障；数字签名技术为在网络通信之前身份相互认证的有效方法，既能在通信过程中保证信息完整性靠手段，又能在通信结束之后防止通信双方相互抵赖。

云计算访问控制

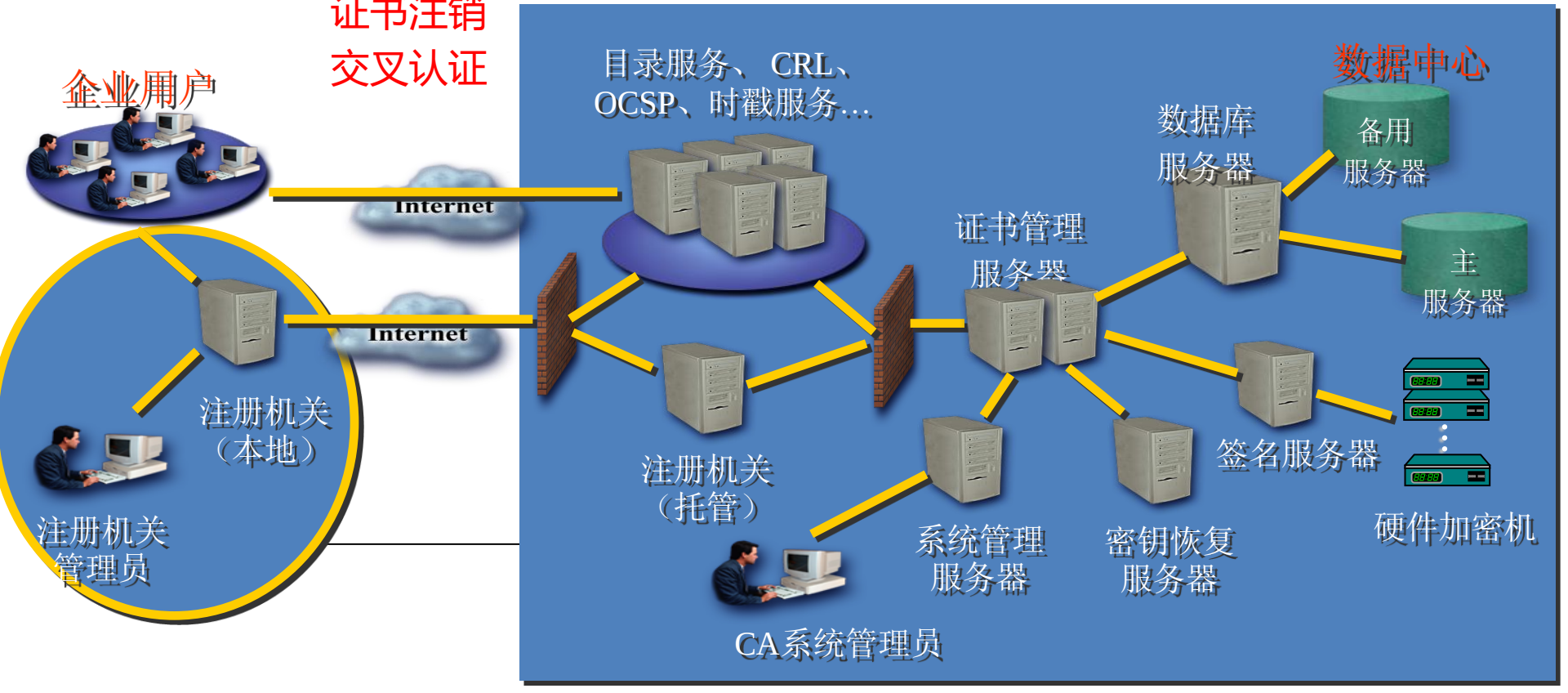
自主访问控制（DAC）：基于身份的访问控制。指一个主体能够自主地把一个客体的一种访问权限及多种访问权限授予给其他主体，并且能够对这些授权予以撤销，只要该主体拥有这个客体。其基本思想是基于对主体的识别来限制对客体的访问，自主的含义是指系统中的主体能够自主地把其拥有的对客体的访问权限全部或部分地授予给其他主体。DAC提供的安全性较低，无法对系统资源提供严格保护。

强制访问控制（MAC）：基于规则的访问控制。指一个主体必须经过系统授权才可决定其是否能够对客体进行访问，以及进行什么层次的访问。这种访问控制机制是对主体和客体分别进行安全的标记，而且有访问请求时，比较主体与客体的安全标记，用来决定主体是否拥有权限访问客体。

基于角色的访问控制（RBAC）：将访问许可权分配给一定的角色，用户通过饰演不同的角色获得角色所拥有的访问许可权。RBAC基本思想是将对客体的访问权限赋给角色，再将角色赋给用户。并且角色和用户以及权限并非单一的关系，一个用户可以被指派给多个角色，同一个角色也可以被指派给多个用户，一个角色可以有多个权限，一种权限也可以分配给多个角色。用户要进行访问就必须建立起一个会话，一个会话一次只能对应一个用户，但是一次会话中的用户可以根据自己的需要来动态激活自己拥有的角色，从而拥有激活角色的所有权限。在RBAC中，通过分配和取消角色来完成用户权限的授予和撤销。实现了用户与访问权限的逻辑分离，极大地简化了权限管理。

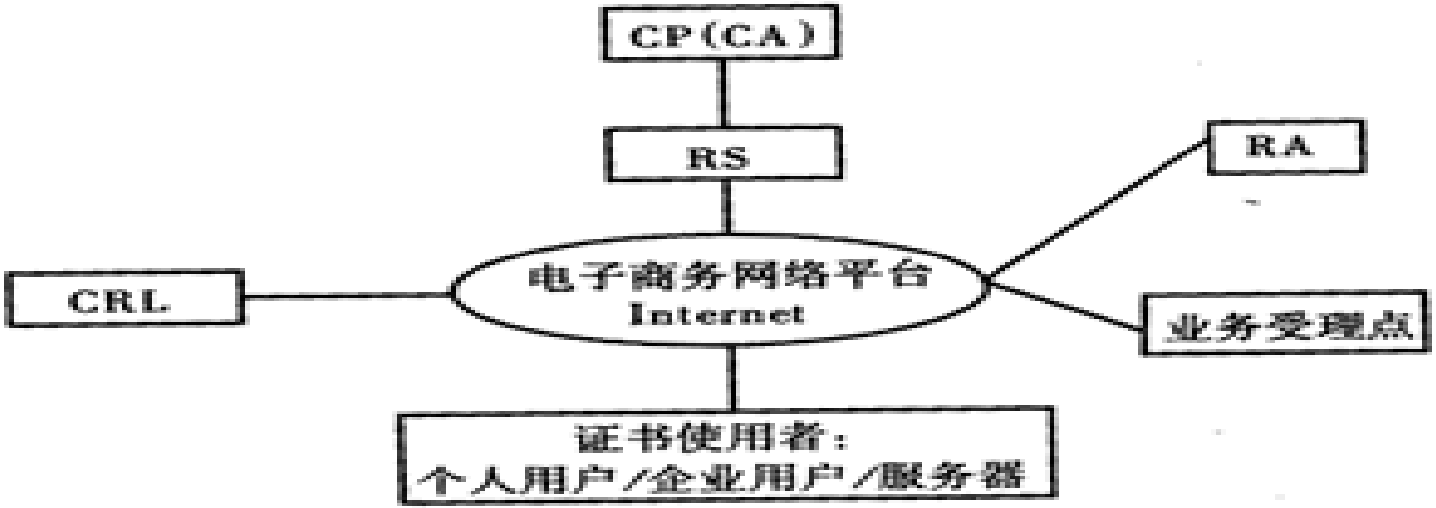
PKI/CA

认证中心CA
数字证书库
密钥的备份、恢复系统
证书注销
交叉认证

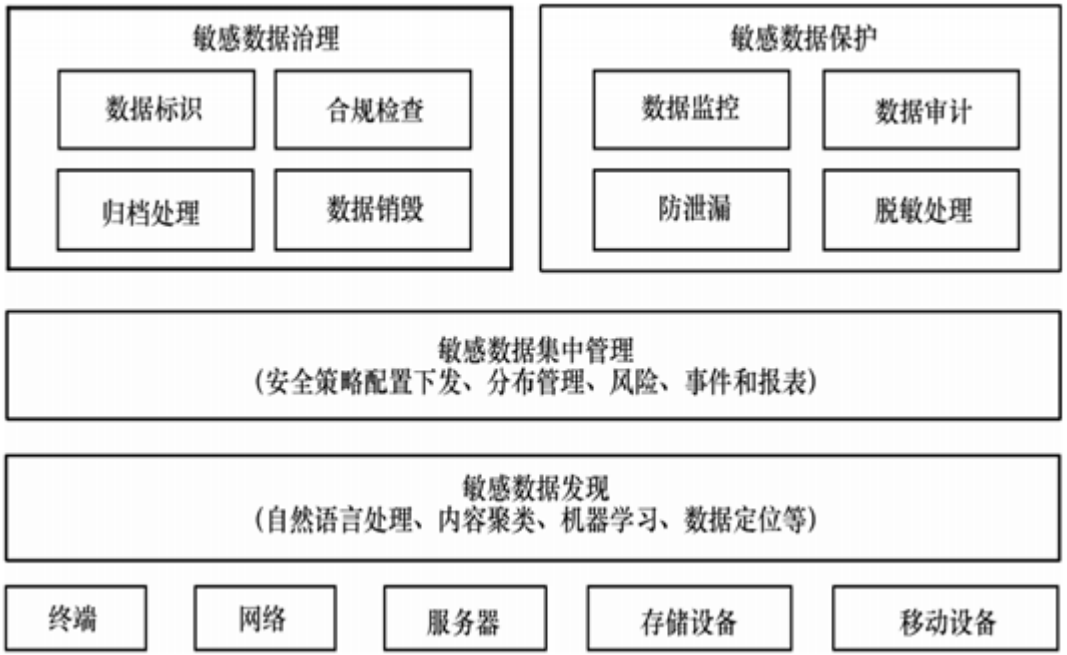


PKI/CA

CA认证体系从功能模块来划分，大致可分为以下几部分：接收用户证书申请的证书受理者（RS）。证书发放的审核部门（RA）。证书发放的操作部门（CP），一般称这部分为CA以及记录作废证书的证书作废表（CRL）



云计算安全技术



敏感数据保护：

敏感数据发现层包括敏感数据发现算法和保护代理模块，在系统中的主机、服务器、数据库等部署敏感数据安全代理，实现基于内容识别技术的敏感数据发现，敏感数据安全保护策略的接收和执行。

敏感数据集中管理层是在敏感数据发现的基础上，建立起云环境下敏感数据总体分布视图，进行统一的敏感数据保护策略的管理和下发，通过安全策略配置，实现差异化的敏感数据治理和保护机制。

敏感数据治理和保护层是下层安全策略的执行层。敏感数据治理是指敏感数据的安全管理，主要包括数据标识、合规检查、数据归档和数据销毁等数据安全管理要求；敏感数据保护是指敏感数据的安全技术保护手段，主要包括数据脱敏、敏感数据使用监控、敏感数据防泄露和敏感数据审计与追溯等安全措施。

云计算安全技术

数据脱敏：根据数据保护规范和脱敏策略，对业务数据中的敏感信息实施自动变形， 实现对敏感信息的隐藏。 数据脱敏的内涵是：借助数据脱敏技术，屏蔽敏感信息，并使屏蔽的信息保留其原始数据格式和属性，以确保应用程序可在使用脱敏数据的开发与测试过程中正常运行。

脱敏算法的设计与数据应用的场景紧密相关，设计脱敏算法时要考虑的因素：

1 脱敏后保持数据业务属性，取值范围合理

业务数据中的人名、地名、日期等在脱敏后需要保持可读性。 脱敏后数据要能正确通过有效性验证，如身份证号的校验码和生日区间，取值范围要合理，如信用卡号变换后仍属于发卡行号码区间等。

2 脱敏后保持数据完整性， 保留数据字段原格式长度不变

如对手机号码进行脱敏，脱敏完成后仍为11位手机号码，并且已经不是真实的电话号码。脱敏可逆性，脱敏后可以恢复成原始业务数据。 随着大数据分析应用的逐步开展，业务部门经常需要将脱敏后的数据还原成原始业务数据，以开展下一步工作。

3 脱敏可重复性，不同轮次的脱敏结果相同

例如，某企业正在内部测试一套分析预测系统，为了更好地 再现真实场景，需要连续从业务系统每周抽取客户购买数据，要求每次抽取脱敏过程对同一身份证号 的变换结果必须相同，才可以保证同一客户的行为模式被正确理解分析，否则就会影响测试效果。 在 另外的场景中，出于保密考虑，对同一字段每次进行脱敏的结果都要求具有一定差异性，以避免黑客 或内部员工收集大量脱敏前后数据，通过逆向工程 还原真实数据。

4 脱敏完备性，防止使用非敏感数据重建敏感数据

如果某些非敏感数据的组合可以用来重建敏感数据， 那么这些非敏感数据也需要进行脱敏处理。

脱敏算法对比

技术类别	主要优点	主要缺点	代表技术	典型应用场景
基于公开加密算法的技术	数据真实、无缺损； 高隐私保护度、可逆、可重复	计算开销大； 部署简单	公开对称加密算法如3DES,AES,SM1,SM4等	在分布式环境下的通信加密； 机密性要求高； 不需要关联性； 不需要保持业务属性
基于数据失真的技术	计算开销小；实现简单	数据失真，不可逆	随机干扰； 阻塞、凝聚； 不可逆乱序； 散列如 MD5/SHA1	群体信息统计； 需要关联性； 需要保持业务属性
	适用于各类数据、众多应用，算法通用性高； 能保证发布数据的真实性，实现简单	存在一定程度的数据缺损； 存在一定程度的隐私泄露； 不可逆； 算法成熟度低，目前应用范围不广	匿名化模型： · k-匿名； · l-diversity； · m-invariance 匿名化算法： · Mondrian； · Incognito； · r-cellular	群体信息统计
基于可逆的置换算法	数据真实、无缺损； 算法效率高； 可逆，可重复	算法规则或映射表泄露对隐私数据威胁较大	位置变换； 表映射变换； 基于算法的映射	需要保持业务关联性； 需要可逆的厂商； 需要保持业务属性

IaaS

IaaS即基础设施即服务。通过虚拟化技术整合计算、存储和网络资源形成资源池，在互联网上以服务的方式按需进行弹性的分配。
IaaS的服务理念：将硬件和基础软件，通过网络以服务的形式交付给用户，用户可以在这个平台上安装部署各自的应用系统。

IAAS服务通常包括：

- 服务器设备提供的计算服务
- 数据存储空间提供的存储服务
- 网络和通信系统提供的通信服务
- 操作系统、通用中间件和数据库等软件服务

IaaS

- IaaS以服务模式提供计算、存储、网络等基础设施资源给用户。传统方式下企业需要去买物理服务器、存储等硬件来承载本地应用，让企业业务运行起来。通过IaaS，企业可将硬件外包给IaaS供应商，供应商会提供可支撑企业应用的服务器，存储和网络硬件及虚拟化软件，对上层业务提供虚拟机或其他基础设施资源。
- 使用者能够部署和运行任意软件，包括操作系统和应用程序。
- 使用者不管理或控制底层云基础设施，但可以控制操作系统、存储、已部署的应用程序，并可能限制对某些网络组件(例如主机防火墙)的控制。
- Iaas平台的产品有：OPENStack（Rackspace和NASA联手推出的云计算平台）和Cloudstack。
- 主流的IaaS供应商包括Amazon EC2、Microsoft Azure，VMWare、GoGrid、iland、Rackspace云服务器、ReliaCloud、AliYun、腾讯

IaaS

在复杂分散异构混合的IT环境下，基础设施可能包括传统的物理设备、主流虚拟化平台、私有云平台、容器云平台等。为保障IT基础设施的安全和合规，必须要构建对于异构基础设施的统一管理能力，包括对于不同基础设施的灵活接入，基础设施自身的监控、运维以及容量规划、安全策略、全生命周期管理、门户自服务等等。

IaaS

- 快速服务交付：服务一般整合计算、存储、网络、安全等资源，以服务目录的形式打包提供给用户(应用维护人员或最终用户)。在设计服务目录时尽可能屏蔽技术复杂性，如服务目录中提供的存储一般要整合数据备份功能，以简化云服务使用者的数据管理复杂度。
- 当用户需要某一服务时，只需从模板库中选择自己需要的功能，发出申请，云计算平台会自动按用户的需求在几分钟之内就构建成，能够为服务的使用者提供极大的敏捷性和便利性。
- 虚拟化：虽然云数据中心会保留某些传统应用的物理设备，但虚拟化在提升资源利用率、物理设备与逻辑服务解耦方面是必需的技术，也是组件标准化的最重要实现方式之一。使用虚拟化技术或容器技术来提高硬件资源利用率，服务轻量化。加快应用上线效率。

IaaS

- 集中化与自动化：云计算平台是通过最新的IT技术对传统数据中心的升华，最集中的体现是通过虚拟化技术引擎和云计算管理平台，实现对资源的统一化及动态化分配和管理。传统物理服务器、云计算服务器、网络和存储通过虚拟化技术，可以看作是一个超大规模的计算机，好像只有一个服务器、一个网络设备、一个大的存储。通过管理平台，可以对其进行资源的动态分配和调整及回收。
- 高可靠：云计算管理平台都可以自动发现任何一台服务器的失败，并把失败的服务器从可用服务器列表中剔除，从而保证任意时间用户请求的计算资源都是建立的可用的服务器之上。云计算管理平台可以将失败服务器上的负载自动迁移到可用的服务器上，保障应用负载在硬件失败时自动恢复。云计算管理平台提供虚拟的负载均衡器。负载均衡器把用户请求转发到多台应用服务器上，达到高可用性、负载平衡的运行环境。

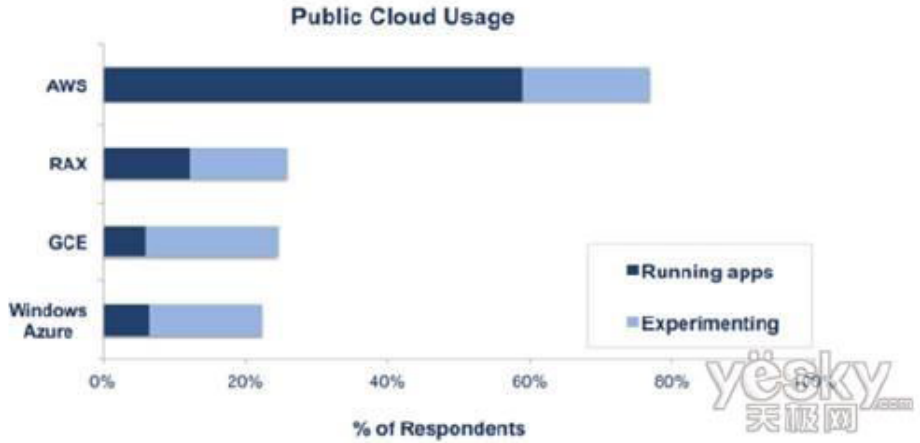
IaaS

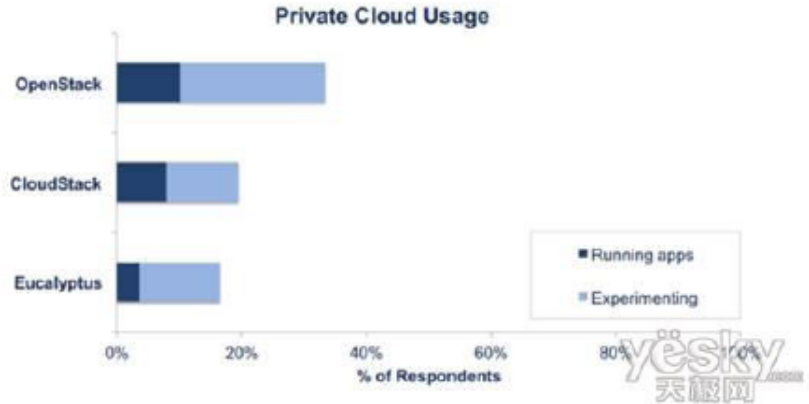
- 标准化：云数据中心标准化包含基础设施标准化、服务组件标准化、服务接口标准化、应用部署标准化等，标准化是云数据中心的基石。
- 资源弹性调度与扩展能力：根据业务应用需要自动分配及回收计算、存储等IT资源，在不同的计算资源甚至数据中心之间动态调度业务应用流量。计算能力不足时，可以在线增加计算服务器，系统自动安装，自动加载到管理平台中心，用作新的计算节点。存储不足时，可以在线增加存储服务器，提供新的存储容量的需求。
- 资源计量和成本核算：资源计量计费是云数据中心对其所提供的服务根据使用者的使用情况进行度量，在数据中心中，资源计量计费可以根据业务或应用对云数据中心资源使用情况进行评估并进行成本核算。避免资源因闲置而浪费。
- 数据中心物理布局的“去中心化”：以往云数据中心在物理布局上严格区分主活、备份、灾备等角色，现在不再强调数据中心分级管理，而是主张在IT资源统一调度前提下，根据物理布局将数据中心划分为不同的可用域，并规范应用开发和部署，将不同的数据和应用按延时、可用性要求布局在不同的可用域中

IaaS领域的几个主角

亚马逊的成功与模仿者

- 亚马逊公有云占据了80%的市场，遥遥领先
- **其API定义已成事实标准，其架构设计被广泛模仿**

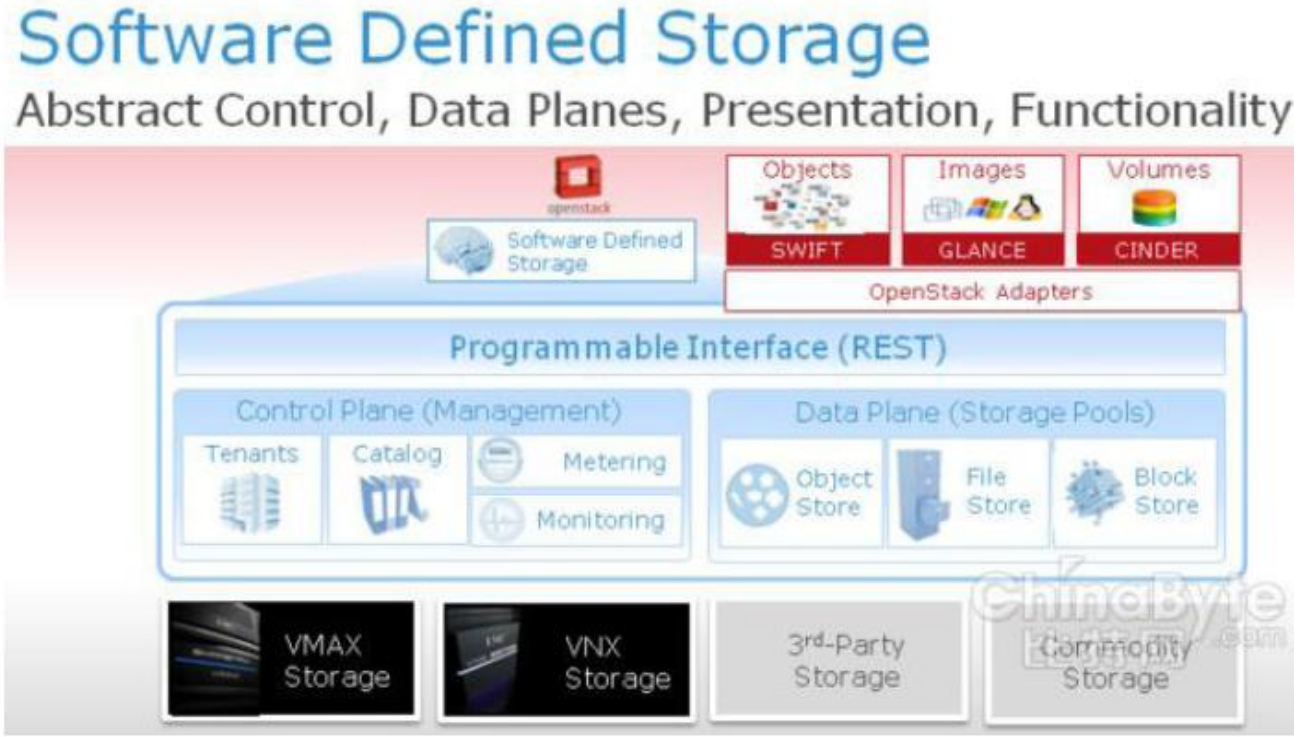




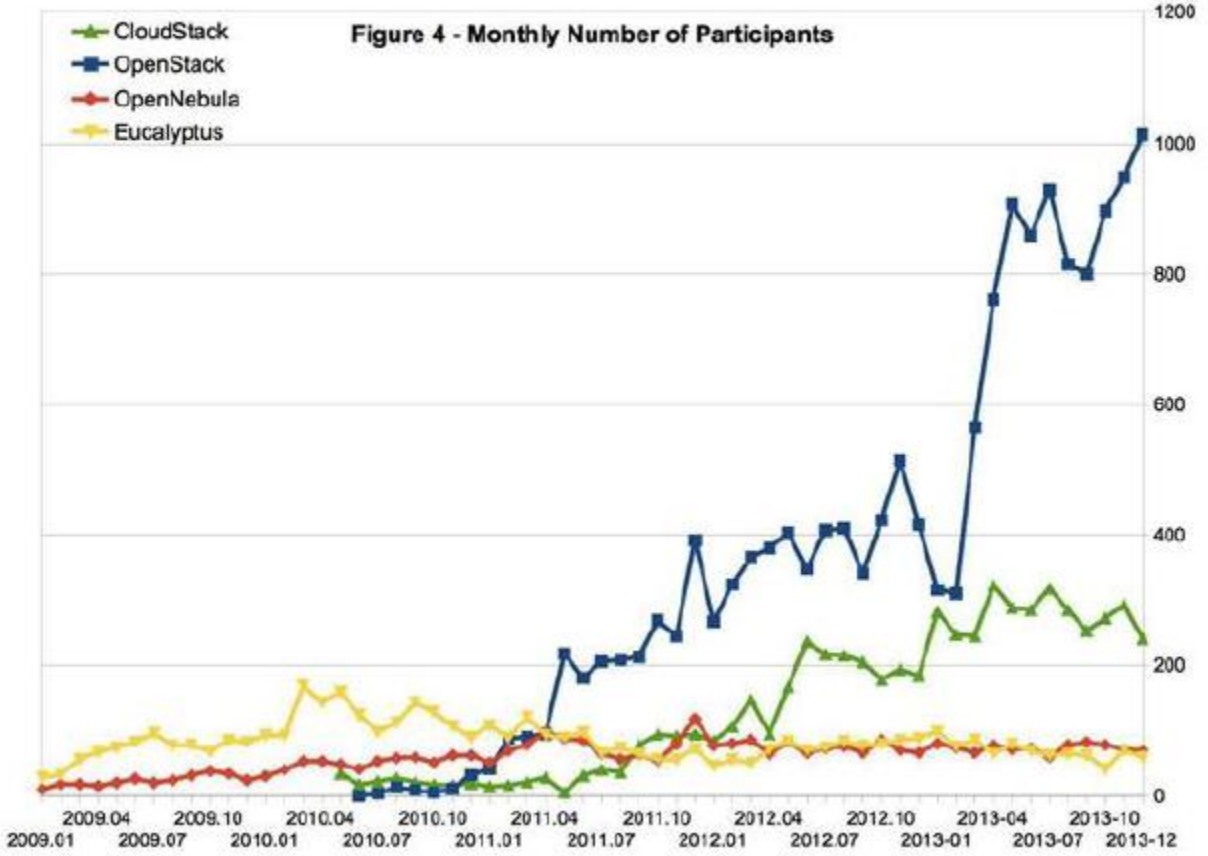
- Eucalyptus ： 是最早试图克隆AWS的开源IaaS云平台
- CloudStack ： 在2010年已经有相当多的运营商和企业客户，2011年7月，Citrix花费2亿美金收购了Cloud.com。2012年四月捐献给Apache，彻底开源
- Openstack：2010年7月份NASA和Rackspace公司将其开源时已经获得25个企业和组织的支持。 2011年7月HP加入、 2012年4月IBM加入OpenStack项目。

VMWare的野心

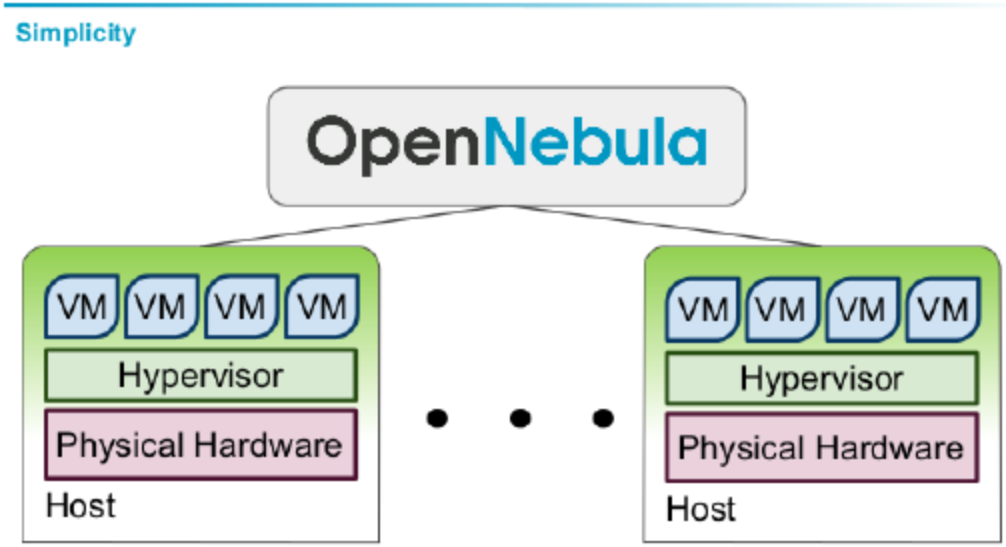
软件定义一切



开源IaaS平台胜负已成定局

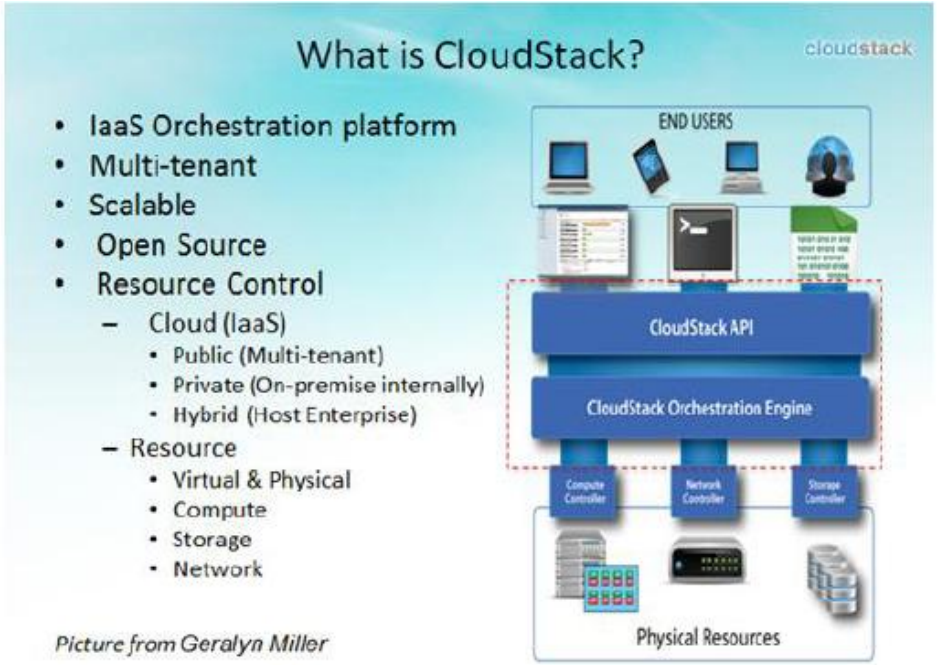


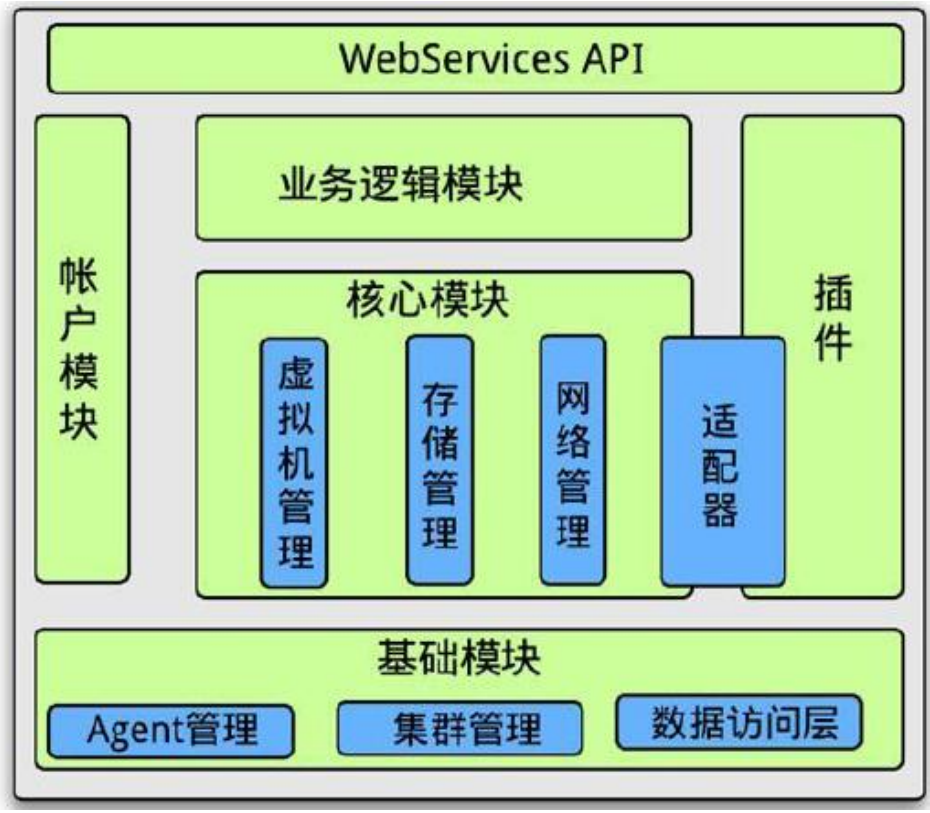
OpenStack vs OpenNebula



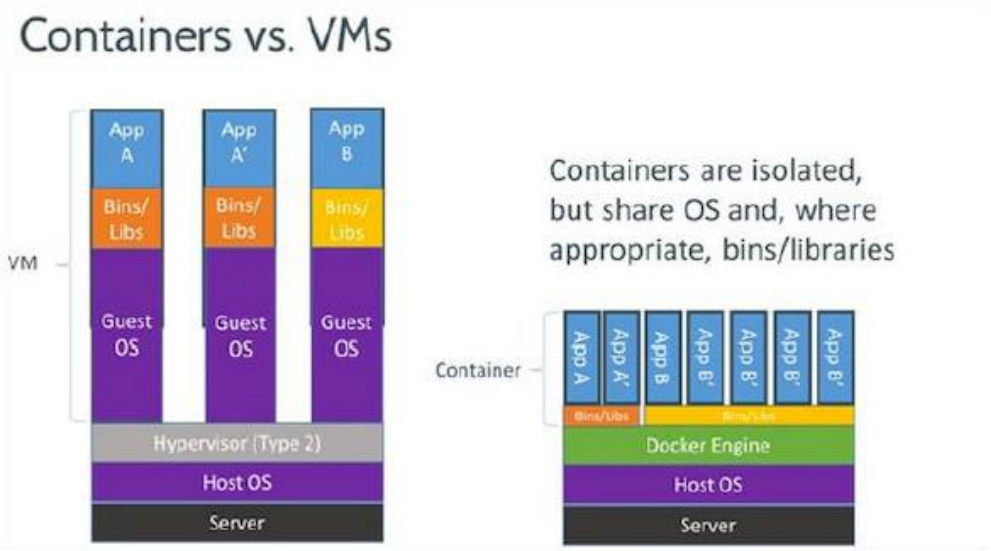
Nebula的开发原则就是轻量，简单。
所以除了在控制节点运行几个接受用户请求的服务外，在计算节点不需要运行任何服务

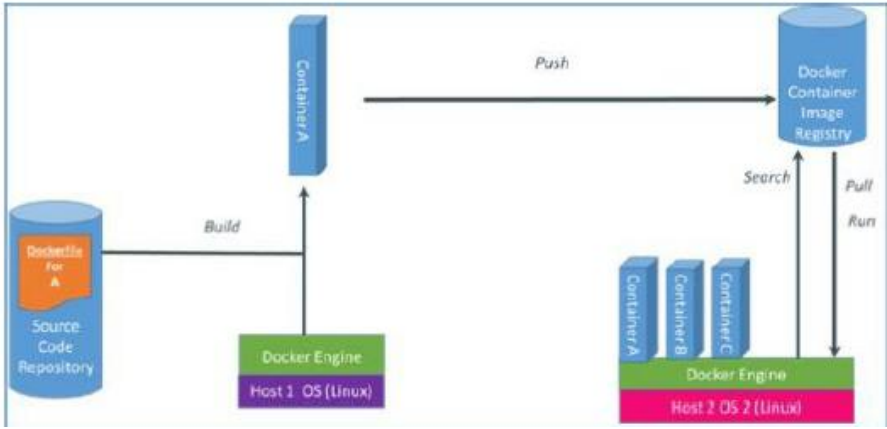
OpenStack vs CloudStack





Docker的崛起与CoreOS的分手事件



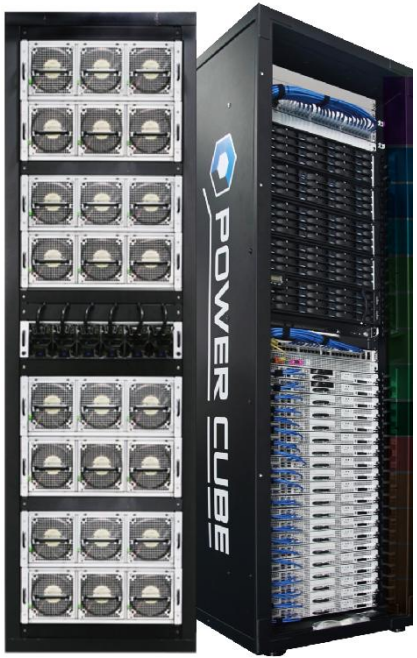


网络基础设施

- 云计算服务的实施有两个步骤，一是将各种计算资源进行整合，二是将应用和服务提供给最终用户。这两个步骤都离不开网络基础设施。
- 网络基础设施使得云计算服务的最终用户不再需要理解交换机、路由器、网络优化设备的工作方式。提高资源利用率，减少业务需求的响应时间，降低运营管理的复杂度，为云计算的应用提供了可靠保障。

云计算机柜管理系统

- 通过整机柜设计，将服务器、存储、交换机、防火墙整合在一个机柜之中。云计算机柜管理系统对每个节点以及机柜系统进行统一管理，达成快速交付，降低能耗，方便运维

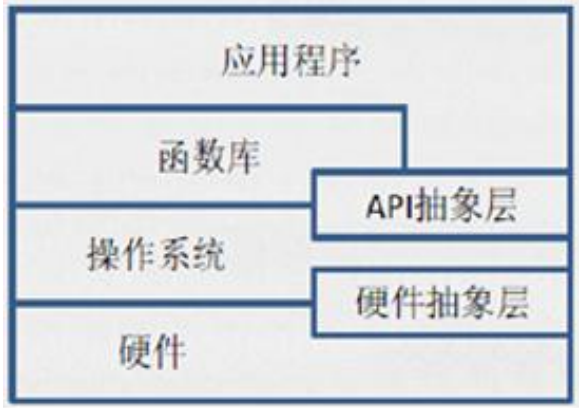


- 机柜规格：44U机柜，服务器、存储、交换机、防火墙融合在一起
- 电源设计：在机柜背面有两个电源总线，6个2700W的电源，为每个节点供电。节点不再配备电源、风扇。
- 风扇设计：机柜背后有4个风扇模块，每个风扇模块有6个热拔插风扇。
- 一个云计算机柜就是一个云计算中心

虚拟化技术

- 计算机系统中的抽象层。虚拟化就是由位于下层的软件模块，通过向上一层软件模块提供一个与它原先所期待的运行环境完全一致的接口的方法，抽象出一个虚拟的软件或硬件接口，使得上层软件可以直接运行在虚拟环境上。
- 20世纪60年代由IBM提出。虚拟化被公认为是云计算的基础技术。虚拟化是一个广义的术语，在计算机方面通常是指计算元件在虚拟的基础上而不是真实的基础上运行。虚拟化技术是将硬件资源或较低层的应用抽象出来，为系统、上层应用或最终用户提供服务的技术。
- Bootload加载操作系统。《计算机组成原理》和《操作系统》

1.每一层都向上一层提供一个抽象接口:
2.每一层只需知道下一层的抽象接口即可，不需要知道其内部运作机制;
3.降低系统设计复杂性;
4.提高软件可移植性:



无虚拟化情况

虚拟化技术

西游记第二十三回。三藏不忘本，四圣试禅心。唐僧师徒四人一路西行。一日天色已晚，见前面一户人家，便入户借宿。闻得这户家财万贯，三个女儿长得如花似玉，想招夫立家。唐僧听后吓得连叫“阿弥陀佛”；悟空不喜此道，便也不放在心上；沙僧一心跟随师父取经，也不愿留下；惟独八戒起了色心，急着去见三个女子。见了老妇人连声叫娘，央求招他为婿。老妇人让他与三个女儿撞天婚，撞着哪个就与他婚配。八戒在三个女儿中左迎右抱，磕磕碰碰，结果一个都没撞着，懊恼不已。

第二天醒来，师徒才发现，原来自己昨晚睡在树林之中，根本没有什么漂亮的房屋，好色的猪八戒被紧紧地绑着，倒挂在一棵树上。大家这时才明白是菩萨在试他们的诚心。

虚拟化的好处

- 虚拟化所实现的计算资源池化、动态调配、自动化管理、高可用等功能是云计算平台所依赖的必不可少的基础特征；虚拟化和云计算的基础硬件平台一定是x86。小型机和大型机无论在动态伸缩性、灵活性、兼容性还是在性价比方面不适合作为云计算的硬件基础平台。
- 通过虚拟化技术，系统或者上层应用或最终用户可以不去了解资源的实际组织方式。
- 能够充分利用各种资源。如利用虚拟化技术，一台计算机可以同时运行多个操作系统，而且每一个操作系统中都可以有多个程序运行。这些程序实际上都运行在同一个CPU上，从而使主机的硬件资源得到充分利用。
- 扩大硬件的容量，简化软件的重新配置过程。CPU的虚拟化技术可以单CPU模拟多CPU并行，允许一个平台同时运行多个操作系统，并且应用程序都可以在相互独立的空间内运行而互不影响，从而显著提高计算机的工作效率。
- 用来测试系统、隔离病毒、进行资源分配
- 虚拟化技术出现以后被迅速应用到计算机的各个组成部分上。包括内存、外存、处理器、软件、网络等。典型的虚拟机软件（事实标准）有Citrix Xen、VMware ESX Server和Microsoft Hype-V等。

虚拟化的优点和缺点

- 封装(逻辑化)

-快照、克隆、挂起、迁移

- 多实例

-计算资源的充分利用率、绿色节能、降低成本

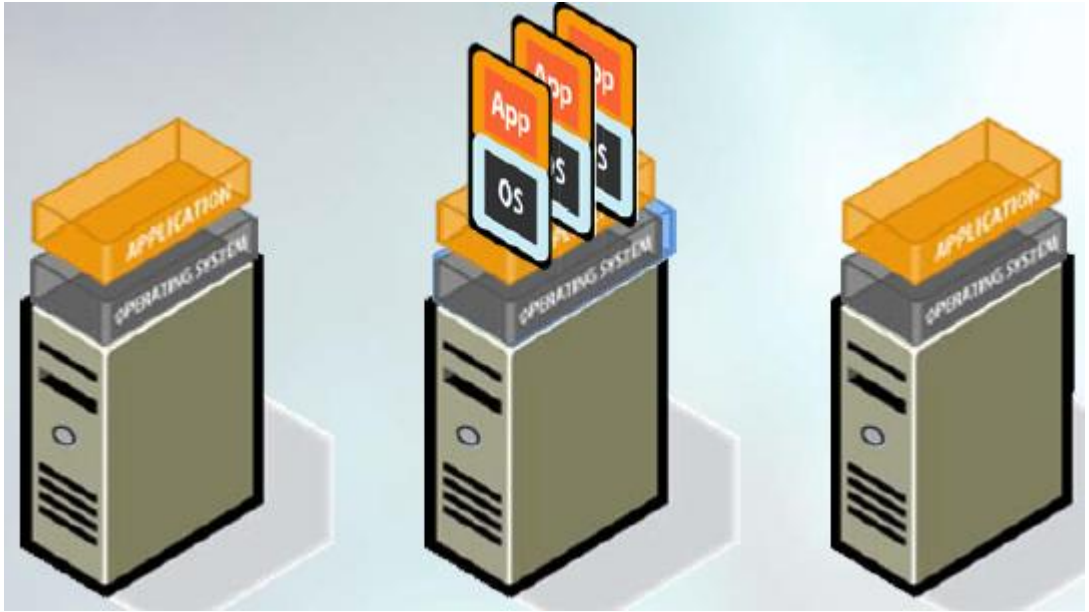
- 隔离
- 硬件兼容
- 虚拟化层特权

-入侵检测、病毒防护、细粒度I0控制

- 性能:虚拟化是对计算资源的分割;
- 错误:虚拟化层的引入增加了系统出错层面;
- 安全:虚拟化会带来- -些安全隐患;
- 影响:一台服务器宕机会影响其上所有虚拟机;
- 复杂:带来管理上的复杂性;

虚拟化技术

- 虚拟化技术主要解决如何让软件应用与底层硬件实现分离。例如X86的机器能够运行Mac OS系统以及应用。让同一台物理服务器可以运行多个应用系统，从而提升物理服务器的利用率。例如原来每个机器的利用率只有20%，但各自运行着Windows Server以及Linux，通过虚拟化技术使得中间的服务器能够同时承载不同的OS，从而可以运行不同的应用，提升利用率（中间服务器利用率达到70%），降低能耗（两边的服务器可以关机）。并且可以实现不同应用的数据隔离。



云就是箱


We're all in.
SQL is in.
is in.
Windows is in.
Exchange is in.
Microsoft

虚拟化监视器VMM

- VMM是一个系统软件，维护多个高效的、隔离的程序环境，支持用户直接去访问真实硬件，而这样的程序环境就称为虚拟机。虚拟机是一个真实存在的计算机系统的硬软件副本，其中部分虚拟处理器指令子集以本地(native)方式执行在宿主(host)处理机上，其他部分指令以仿真方式执行。从以上定义可以看出，VMM管理计算机系统的真实资源，为虚拟机提供接口。使用VMM 有以下优点：
- a)VMM实现相比于Linux或Windows这类操作系统的实现要简单很多。因为VMM避免了像TCPIP Sockets和文件系统这类高级抽象，这将有利于安全性和可靠性，也便于扩展和修改。
- b)VMM允许系统管理者配置虚拟机运行的环境。虚拟机的各项设置可以与真实机不同，如真实机有512MB内存，可以设置虚拟机内存64MB，有利于开发者在各种环境下测试软件。
- c)VMM允许在相同硬件上同时执行不同的操作系统，称为GuestOS。系统管理者可以用这种能力来联合多个使用不充分的分散计算机，为不可信和不安全代码增强了隔离性，同时增强了可靠性，在一个虚拟机中的软件发生故障也不会影响到其他虚拟机。
- d)当操作系统升级后，仍然可以在虚拟机中运行早期开发的软件，由此可以降低软件开发成本。同时成本的降低还来源于减少硬件产品的购置。
- e)针对拥有10～100个处理器的可扩展计算机，VMM能够方便地开发功能强大、可靠的系统软件。
- f)虚拟机控制了程序运行的整个软件环境，包括操作系统和应用软件，因此可以封装程序地址

虚拟化监视器VMM

管理虚拟环境　VMM(virtual machine monitor)

-虚拟资源

- 处理虚拟化模块
- 内存虚拟化模块
- 设备虚拟化模块

-虚拟环境调度

- 虚拟处理器上下文调度

-虚拟机间通信

- 特权域与虚拟机之间的通讯
- 普通虚拟机之间的通讯

-虚拟环境管理接口

- 为用户提供管理界面

- 管理物理资源

-处理器管理
-内存管理
-中断管理
-系统时间维护
-设备管理

- 其他模块

-软件定时器
-电源管理
-安全机制
-多处理器同步原语
-性能采集和分析工具
-调试工具

虚拟化监视器VMM分类

- 按照虚拟平台分为全虚拟化（Full Virtualization） （虚拟的平台和现实的平台一样 ）和半虚拟化（Para Virtualization）（虚拟的平台和现实的平台不一样）
- 1）全虚拟化：无需对运行在虚拟化平台上的OS做任何修改；
- 1，软件辅助的全虚拟化：常见做法是优先级压缩（Ring Compression）和二进制代码翻译（Binary Translation）。
- 优先级压缩： 让guest OS跑在较VMM的ring 0 低的优先级ring 1 上，当需要执行特权指令时就触发异常，被VMM截获。
- 二进制代码翻译：VMM对于guest的二进制代码，发现需要处理的guest指令就将其翻译成支持虚拟化的指令。
- 2，硬件辅助完全虚拟化：软件辅助的虚拟化，就相当于在系统上打补丁，x86厂商尝试在硬件的层面来改善这个问题，就是在硬件本身加入足够的虚拟化功能，可以截获操作系统对于敏感资源和敏感指令的操作，从而向VMM报告异常。比如intel的VT-x， 在处理器中引入一个特殊的模式，操作系统一旦进入该模式，无法察觉该模式，但是任何操作都会被该模式报告VMM。
- 2）半虚拟化：在源码级别修改指令以回避虚拟化漏洞，简单来说，就是运行在虚拟化平台上的OS是被动过手脚的。典型的做法就是修改OS的相关处理器代码，让出ring0 ，或者定制指定的I/O协议，以期提高读写效率。

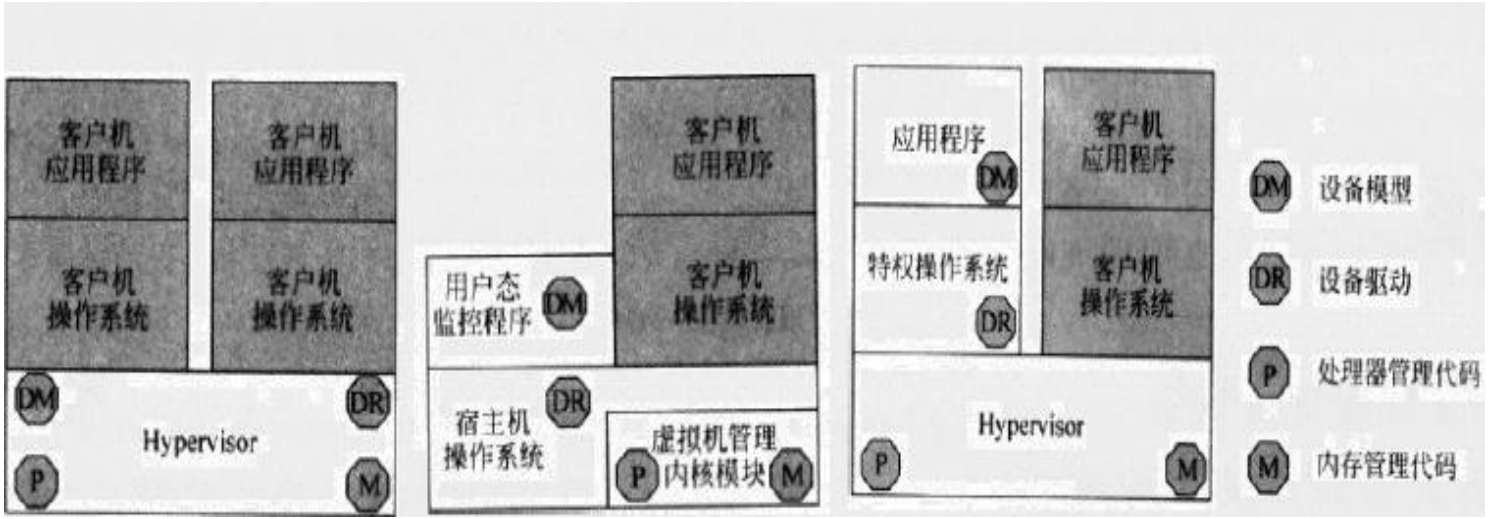
虚拟化监视器VMM分类

- 按照VMM的实现架构分为Hypervisor模型、宿主模型和混合模型。
- 1）Hypervisor模型：VMM可以视为一个具有虚拟化功能的操作系统，即管理物理资源和虚拟环境的创建、管理。
- 优点： 效率高； 缺点：只支持部分型号设备，需要重写驱动或者协议。
- 典型产品：VMware ESX server3， KVM
- 2）宿主模型：宿主机OS管理物理资源，VMM作为宿主机OS的一个独立的内核模块来提供虚拟化功能。VMM通过调用宿主OS的相关服务来获取资源，创建出来的虚拟机也作为宿主OS的一个进程来参与调度。
- 优点：个人理解就是充分利用现有的OS的device driver，无需重写；物理资源的管理直接利用宿主OS来完成。
- 缺点： 效率不够高，安全性一般、依赖于VMM和宿主OS的安全性。
- 典型产品： VMware server ， VMware workstation，virtual PC， virtual server，
- 3）混合模型：上述两种模型的混合体。
- VMM处在最底层，拥有全部物理资源，但是与Hypervisor模型不同的是，大部分I/O设备是由一个运行在特权虚拟机中的特权OS来管理的。CPU和Memory的虚拟化依然由VMM来完成，而I/O的虚拟化则由VMM和特权OS来共同完成。混合模型集合了上述两种模型的优点，但是缺点是经常需要在VMM与特权OS之间进行上下文切换，开销较大。
- 典型产品： window server 2008之hyper-v， Xen，

VMM分类

- 按VMM实现结构分类

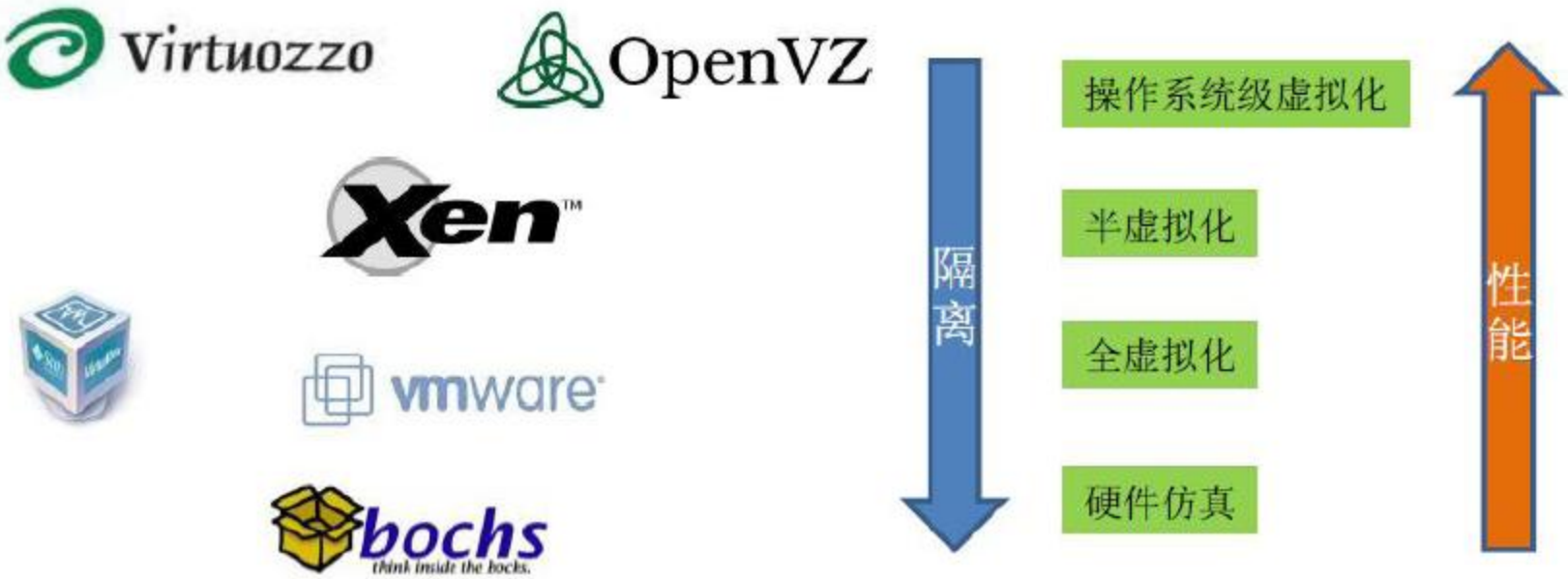
-Hypervisor模型
-宿主模型
-混合模型



虚拟化管理平台VMM

- 提供完善的高可用、安全性、扩展性与自动化管理等功能；管理大量物理和虚拟设备，具有不停机的自我扩展和自我愈合功能，保证上层云计算平台根据业务需求进行资源的动态伸缩调整，保证对云计算消费者的服务质量SLA

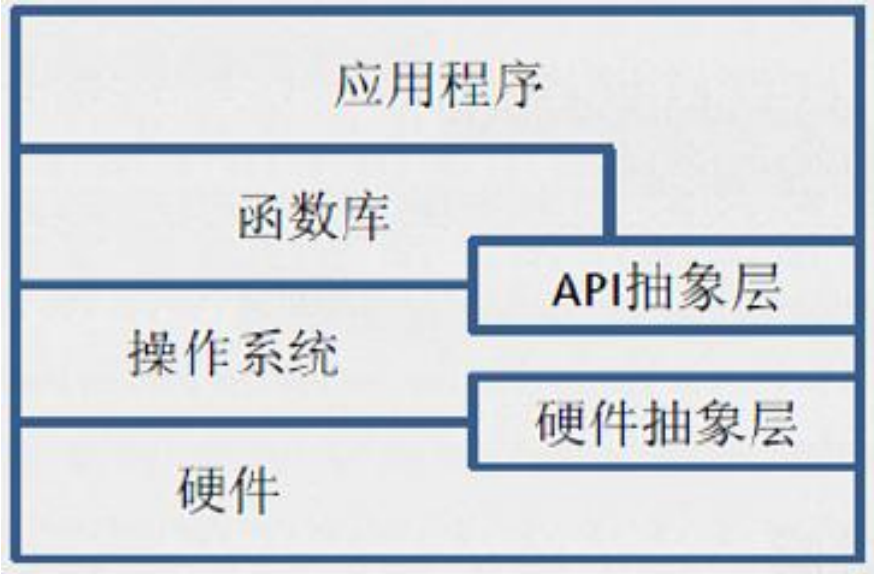
虚拟化技术


Virtuozzo
OpenVZ
Xen
vmware
bochs
think inside the bocks.
隔离
操作系统级虚拟化
半虚拟化
全虚拟化
硬件仿真
性能

虚拟化分类

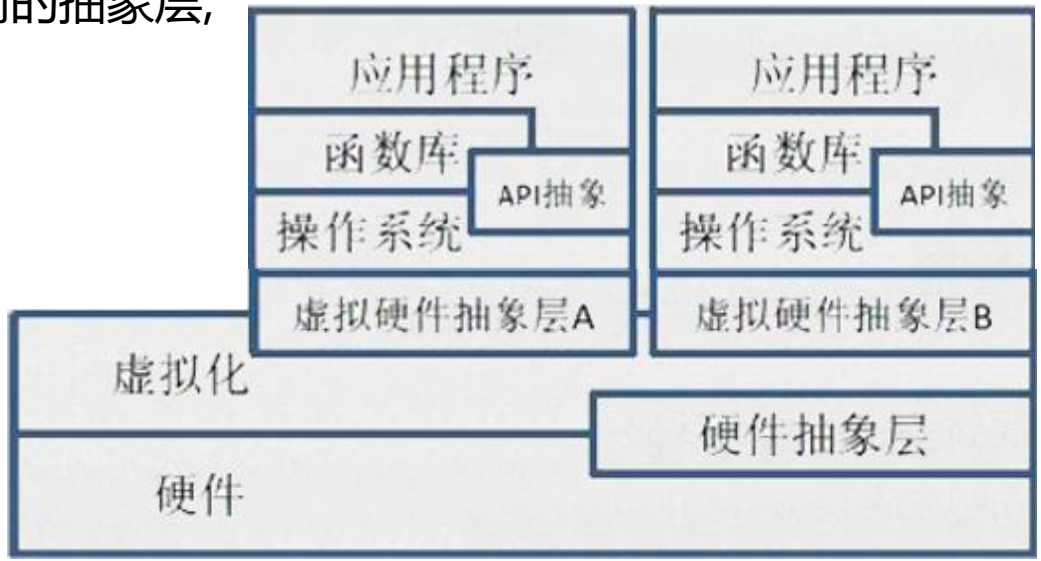
- 硬件抽象层上的虚拟化
- 操作系统层上的虚拟化
  （全虚拟化和半虚拟化）
- 函数库层上的虚拟化
- 编程语言层上的虚拟化

Host（宿主） and Guest（客户）



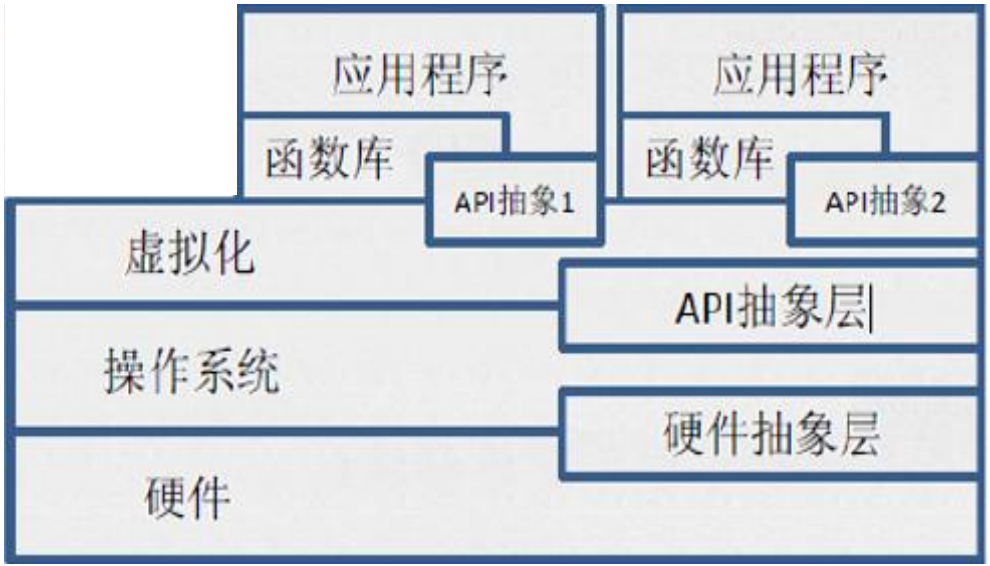
硬件抽象层上的虚拟化

-通过虚拟化硬件抽象层来实现虚拟化，为客户机操作系统提供相同或相近的硬件抽象层;
-相同的指令集;
-特设指令由虚拟化软件进行处理;
-中断控制器、设备等可以是完全不同的抽象层;
-VMWare、Xen



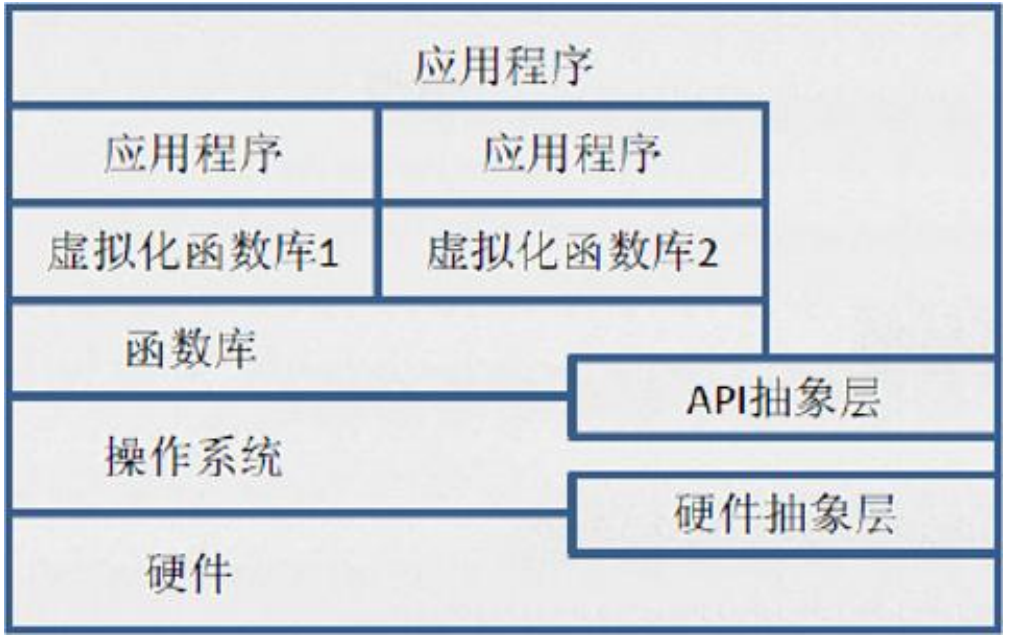
操作系统层上的虚拟化

-操作系统的内核可以提供多个相互隔离的用户态实例(容器)；
-想一想chroot;
-每个容器有独立的文件系统、网络、函数库等;
-灵活性小，Guest OS相同;
-隔离性稍差
-Virtuozzo (VPS) OpenVZ;



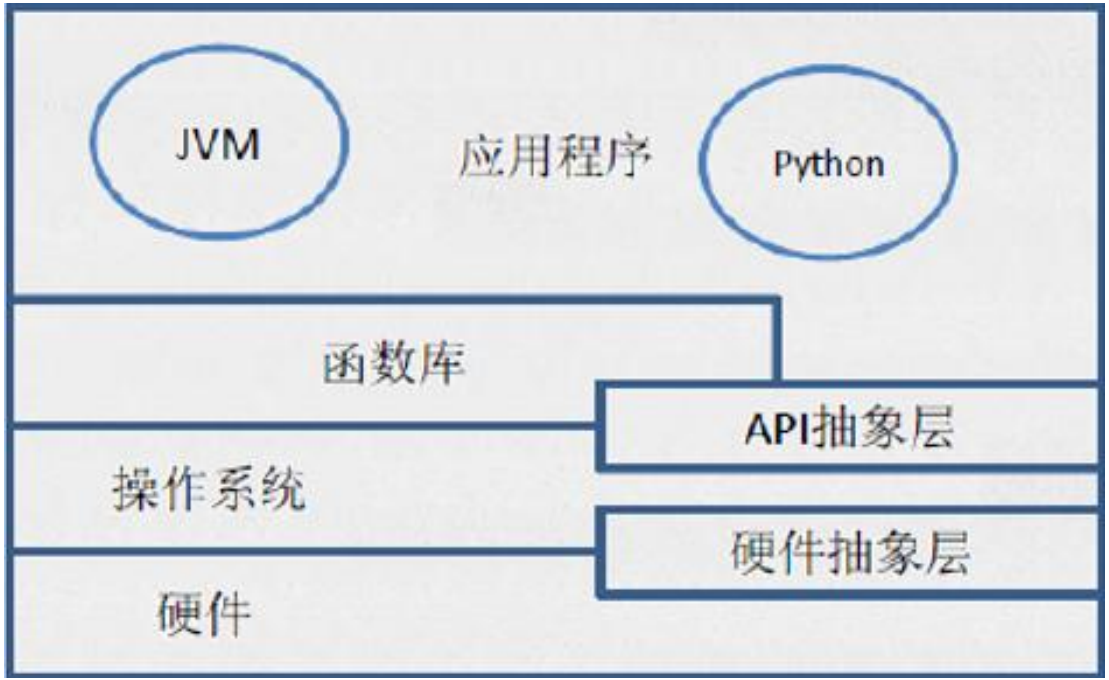
函数库层上的虚拟化

-通过虚拟化操作系统的应用级函数库的服务接口，支持不同的应用程序;
-WINE，在Linux环境下支持Windows程序的执行环境;
-Cygwin,在Windows环境下支持Linux程序的执行环境;



编程语言层上的虚拟化

进程级的虚拟执行环境;
代码翻译为硬件机器语言;
JVM;
跨平台;



谢谢！