云计算技术
课程回顾

中国科学院大学计算机与控制学院
中国科学院计算技术研究所

关于课程

- 课程中文名称：云计算技术
- 课程英文名称：Cloud Computing Technology
- 课程属性：　专业课
- 学时/学分：　40/2
- 预修课程：　计算机组成原理、计算机系统结构、操作系统、计算机网络

教学目标

- 通过本门课程，同学们能够掌握以下内容（基本要求）：
  - **掌握云计算的基本概念、体系结构**
  - **熟悉开源云计算系统的使用（要能用表格进行比较）**
  - **掌握云计算系统的核心组件及其工作原理（要能画出框架图等）**
  - **掌握云计算系统的关键技术及其原理（要能画出原理或过程图）**
  - **了解云计算的发展现状与趋势（）**

云计算定义与内涵

– NIST：云计算是一种能够通过网络以便利的、**按需付费**的方式获取计算**资源（包括网络、服务器、存储、应用和服务等）**并提高其**可用性**的模式，**这些资源来自于一个共享的、可配置的资源池，并能够以最省力和无人干预的方式获取和释放**

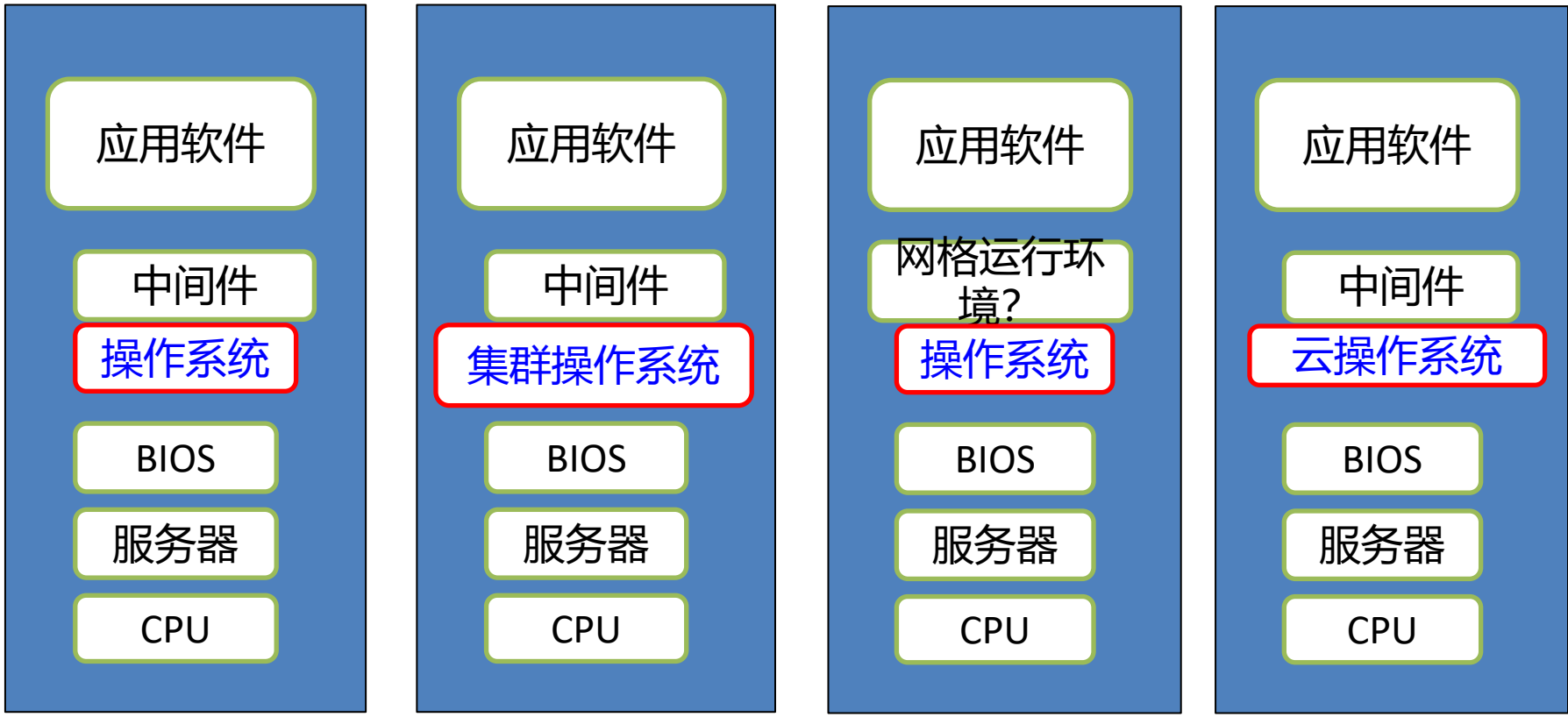
NIST： 美国国家标准和技术研究院

云计算的特点与分类

- 按照部署方式：
  - 公有云
  - 私有云
  - 混合云

- 按照服务模式：
  - 基础设施即服务：Infrastructure as a Service, IaaS
  - 平台即服务：Platform as a Service, PaaS
  - 软件即服务：Software as a Service, SaaS

不同形态的系统在各个层面有不同的特点和需要解决的问题



热闹背后的技术和市场---云计算

云计算的关键技术是对历史技术的继承和发展

- 云计算的技术来源于过去二十年的学术研究
  - 网格技术（grid）
  - 虚拟化技术（virtualization）
  - 面向服务的体系结构技术（SOA与WOA）
  - 公用计算技术（utility computing）
  - 大规模分布式数据处理技术
- 也得益于产业进步
  - 互联网带宽增长，尤其是接入带宽
  - 高性能计算机与网络存储进步
  - 新型互联网浏览器技术进步
  - 各种互联网服务的普及

云计算的技术生态和市场表现

- 技术生态： OpenStack是一个由NASA（美国国家航空航天局）和Rackspace合作研发并发起的，以Apache许可证授权的自由软件和开放源代码项目。
- OpenStack社区参与人员有10万人，参与企业700家以上，提交的代码超过2000万行
- 在Rocky版本中，九州云、**华为**、麒麟云、国内企业挤进Commits全球贡献排名Top10。
- OpenStack 历经9年时间发展，如今已成为全球开源云计算基础设施的事实标准
- 云计算是市场模式驱动的技术
- **市场成长**：“阿里云”，十年一剑，成为国内领跑的公有云服务商；**华为**云计算及鲲鹏生态合作全面突破和落地服务

云计算的关键技术

- 虚拟化技术
- 数据存储技术
- 资源管理技术

云计算的关键技术——虚拟化技术

从物理形态上看，服务器虚拟化的目标是虚拟机

根据虚拟机实现的抽象层次，虚拟化架构可以分为如下几类：

- **指令级虚拟化**
- **硬件级虚拟化**
- **操作系统级虚拟化**
- **编程语言级虚拟化**
- **程序库级虚拟化**

？回顾一下这几类架构之间的区别

- Linux操作系统工作原理
- – 内核态：对应CPU的运行优先级Ring 0（最高），运行在内核态 的程序是Linux Kernel以及Kernel Modules，能够控制所有的硬件 资源（CPU、内存、I/O等），如分配和回收，可以执行所有的 CPU指令，可以访问任意地址的内存，Kernel向应用程序提供的 接口就是系统调用。通过内核控制应用程序对资源的访问，对系 统起到了保护作用，即便应用程序发生了崩溃也不会对整个系统 产生影响。
- – 用户态：对应CPU的运行优先级Ring 3（最低），运行在用户态 的程序是应用程序，在该模式下应用程序没有对硬件资源的直接 控制权限，不能访问任意地址的内存，不能执行所有的CPU指令， 应用程序在调用系统调用的时候实际上调用的是库函数，例如 glibc库，由库函数对内核提供的系统调用进行封装。

云计算的关键技术——虚拟化技术

- 服务器虚拟化
  - 同一组物理资源能够被很多的虚拟主机重复、共享使用，而底层资源的划分以及共享功能的实现都交由虚拟机管理器去完成，然后将虚拟的计算资源提供给上层应用
  - 服务器虚拟化必备的是对三种硬件资源的虚拟化：CPU、内存、设备I/O
  - 为了实现更好的动态资源整合，当前服务器虚拟化大多支持虚拟机的实时迁移。

云计算的关键技术——虚拟化技术

- 服务器虚拟化
  - **CPU虚拟化**
    - **全虚拟机化**：是一种采用二进制代码翻译技术的虚拟化实现，该技术不用修改客户操作系统就能实现全虚拟化技术，但是动态转换指令的步骤将需要使用一定量的性能开销
    - **半虚拟化**：是通过修改客户操作系统实现的，把虚拟化层的超级调用作为特权指令，以此来解决虚拟主机运行特权指令的相关问题

CPU虚拟化

- 硬件级虚拟化
- **临界指令问题**
- CPU虚拟化实现方式——BT+DE
- **CPU虚拟化实现方式——半虚拟化（Xen）**
- CPU虚拟化实现方式——硬件辅助虚拟化
- **KVM VCPU工作原理**
- CPU的工作模式
- X86架构处理器虚拟化“漏洞”

临界指令（Critical Instructions）

- – 非特权指令的敏感指令称为“临界指令”
- – X86 CPU有17条临界指令
- – ARM也存在临界指令
- – IBM S/370、Motolora 68010+、POWERPC等没有临界指令

半虚拟化

- 主要思想：通过修改Guest OS来解决X86架构下的临界指令问题， 同时提升虚拟机的性能
- 不改变应用程序二进制接口（Application Binary Interface, ABI）， 因此应用程序不需要修改和重新编译
- 适合于开源操作系统（Linux、FreeBSD），但很难支持闭源操作 系统，如Windows
- 特权指令、敏感指令相关的操作被转换为发送给VMM的Hypercall （超级调用），由VMM继续进行处理

KVM VCPU工作原理

- KVM虚拟机线程模型
- VCPU的整个生命周期都在Qemu线程的上下文中，在Kernel （root模式）、User（root模式）、Guest（non-root模式）三种 模式下转换（*？结合各模块与工作流程回顾具体的模式和特权级）*
- 每个VCPU对应一个Qemu线程，在Linux上以普通线程进行调度

云计算的关键技术——虚拟化技术

- 服务器虚拟化
  - 内存虚拟化
    - 将服务器的物理内存进行统一管理，为每个虚拟机提供了彼此隔离而连续的虚拟化内存空间。
    - 虚拟机监视器则利用一个虚拟机内存管理单元来维护物理主机内存与虚拟机逻辑内存之间的映射关系

内存虚拟化

内存虚拟化核心问题是地址映射和对应的缺页处理

- 地址映射——直接模式
- **地址映射——影子页表**
- **地址映射——硬件辅助内存虚拟化**

- ***思考三者的区别，各自的优缺点，以及在故障下的处理方式流程***

直接模式

- Guest OS页表项记录的是客户机虚拟地址到宿主机物理地址（机 器地址）的映射
- Guest OS的页表可以直接供宿主机物理MMU部件使用
- Guest OS的页表结构（页表页、页目录页、描述表页）均被VMM 设置为只读，使得Guest OS不能修改自己的页表，更新页表操作 要通过Hypercall调用VMM来完成
- VMM会对页表更新请求进行验证
  - 虚拟机是否试图映射未分配给它的内存页面
  - 虚拟机是否试图修改页目录结构的只读属性

影子页表技术

- Guest OS维护着自身的页表，完成GVA到GPA的映射，该页表并不被宿 主机MMU所使用
- VMM为每套Guest页表维护一套影子页表，对应于Guest中每个进程的页 表，负责将GVA转换成HPA，影子页表对Guest不可见
- Guest OS在进行进程调度切换进程上下文时，会试图将调度进来的进程 的页表基地址写入到CPU的CR3寄存器中，由于读写CR3寄存器的指令 是特权指令，这一操作会导致陷入到VMM，VMM首先保存Guest进程的 页表基地址（GPN），再将该进程对应影子页表的基地址填入到CR3寄 存器中，使得CPU（MMU）能够通过影子页表将GVA直接转换成HPA， 从而完成内存的访问。
- 当客户机读CR3寄存器时，仍然会陷入VMM，从而将之前保存的Guest 进程页表基地址（GPA）返回给客户机。
- 影子页表与Guest进程页表间要保持同步，即影子页表中的页表项的状态/ 权限等要与对应的Guest进程页表项保持一致

硬件辅助的内存虚拟化技术

- EPT（Extended Page Table）技术
  - Guest OS维护自身的页表，完成GVA到GPA的映射
  - 每个Guest对应一套EPT，EPT负责完成GPA到HPA的映射，EPT 由VMM维护，对虚拟机透明
  - Guest地址翻译过程（KVM）
  - EPT页表建立过程

云计算的关键技术——虚拟化技术

- 服务器虚拟化
  - I/O设备虚拟化
    - 将物理机器的真实设备进行统一管理，将其包装成多个虚拟化的设备提供给多台虚拟主机去使用，并能响应每台虚拟主机的设备访问及输入/输出请求。
    - 现在比较常见的输入/输出和设备虚拟化基本都是用软件的方式去实现的。
    - 虚拟化设备的标准化使虚拟主机不需要再依靠底层的物理设备去实现，也便于进行虚拟机的迁移工作

I/O虚拟化的实现方式分类

- I/O指令模拟：通过捕获Guest I/O指令后利用软件方式来模拟虚拟 设备的行为，该方式不需要对Guest OS与驱动程序做修改，具有 良好的兼容性，缺点是性能开销较大
- 半虚拟化：针对I/O指令模拟方式下“拦截-模拟”过程的性能开销 问题，改进Guest与Host间交互方式，提供一种更加直接高效的通 信方式，该方式能够使I/O性能有较大提升，但需要修改Guest OS 与驱动程序，透明性差
- 硬件辅助虚拟化（I/O设备直通/透传技术）：在硬件的辅助下让虚 拟机直接访问I/O设备，而不需要经过VMM的干预。该方式能够提 供最好的I/O性能，且无需修改Guest OS与驱动程序，但限制一个 物理I/O设备只能被一台虚拟机使用。进一步地，为了使物理设备 能够被多台虚拟机共享，PCI-SIG标准化组织发布了SR-IOV （Single Root I/O Virtualization）规范，在I/O设备硬件层面支持 虚拟化。

I/O指令模拟

基于Qemu的虚拟设备I/O一般流程：

- – 1. Guest产生I/O请求，被KVM 截获
- – 2. KVM经过简单处理后将I/O请求存放在I/O共享内存页面
- – 3. KVM通知Qemu，I/O请求已经存入I/O共享内存页面
- – 4. Qemu从I/O共享页拿到I/O请求
- – 5. Qemu模拟本次的I/O操作，并发送给宿主机相应的设备驱动
- – 6. 宿主机硬件完成I/O操作并返回结果给Qemu
- – 7. Qemu将结果放回I/O共享页
- – 8. Qemu通知KVM去I/O共享页拿结果
- – 9. KVM去I/O共享页拿到结果
- – 10. KVM将结果返回给Guest

- 如何加深印象：以网卡、基于USB的打印机、磁盘等设备为例，来画出在Qemu下虚拟网卡等的处理过程

云计算的关键技术——虚拟化技术

服务器虚拟化的特征

(1) 隔离性

- 服务器虚拟化能够将运行于同一个物理主机上的多个虚拟机完全隔离开，多个虚拟机之间的关系就如同是多台物理主机之间一样，每一台虚拟机有着自己相对独立的内存空间。当一台虚拟机崩溃的时候，不会影响到其他虚拟机的正常工作。

(2) 多实例

- 一台物理主机通过服务器虚拟化技术的处理之后，能够运行很多个虚拟服务器，不仅支持多个客户操作系统，而且物理系统的资源还能以可控的方式被分配给各个。

(3) 封装性

- 通过服务器虚拟化处理以后，一个完整的虚拟机环境对外表现为一个单一的实体，这样方便于在不相同的硬件设备间进行复制、移动和备份操作。同时，服务器虚拟化技术将物理主机的硬件封装成标准化的虚拟硬件设备，提供给了每一台虚拟机的操作系统和应用程序，这在很大程度上提高了系统的兼容性。

云计算的关键技术——虚拟化技术

- 服务器虚拟化的优点
  - 快速部署
  - 较高的资源利用率
  - 实时在线迁移
  - 动态资源调度及高兼容性

容器技术---操作系统级虚拟化

- 也称为容器化，是操作系统自身的特性，允许多个**相互隔离（？哪些隔离技术）**的用户空间实例存在，这些用户空间实例被称为容器。
- 容器能解决什么问题？
  - 虚拟机“太重”：性能开销大、内存/存储空间占用高，生产环境下 一台Host上能够运行的虚拟机数量一般在10台左右
  - 启动速度较慢（几十秒钟到数分钟）：启动时需要像在物理机环境下 一样完成BIOS、操作系统以及系统服务的初始化，无法满足需要快速 响应和调整的应用需求
- 思考：为什么虚拟机和容器会同时存在？各自的优缺点？

不得不讲的容器---docker

- docker是一个能够把开发的应用程序自动部署到容器中的开源引擎， 能够使软件开发生命周期高效快速
- OCI（Open Container Initiative，开放容器计划）
  - 容器镜像规范（image spec）
  - 容器运行时规范（runtime spec）

实现了OCI规范的docker

- docker镜像驱动
- AUFS
- OverlayFS
- Dockerfile
- **镜像的分层管理和访问策略**

云计算的关键技术——存储技术

- 存储系统的分类、特点、接口、典型设备或系统以及适用场景
- 分布式存储系统及其类型、CAP理论、一致性策略（NWR策略）
- GFS系统
- HDFS系统
- Ceph分布式存储系统

Ceph分布式存储系统

- 一个统一分布式存储系统 – 一个系统提供不同接口：块接口、文件接口、对象接口 – 降低开发和运维成本
- Ceph通过一组完成不同功能的组件实现了三种结构的访问、本地的接口以及监控运行管理等功能
  - RADOS（Reliable Autonomic Distributed Object Store, RADOS）
  - OSD（Object Storage Device）
  - Ceph Monitor
  - Ceph MDS（Ceph MetaData Server）
  - Librados
  - RBD
  - RADOSGW（RADOS Gateway， RGW）
  - CephFS

Ceph分布式存储系统

- 提供三种结构的访问，每一种接口，当客户端发起读写请求时，客户端定位到数据并获得数据的流程（构图和说明）
  - 涉及到的概念和之间的逻辑关系
  - 具体的流程
  - 对应的数据组织
  - 其它

- Ceph中的数据放置和副本、故障处理

云操作系统

- 基本概念：云操作系统对云计算中心的各种硬件资源进行抽象，封装成API供 云应用调用，提供用户访问云资源与服务的界面
- 云操作系统的两种设计思路
  - 数据中心虚拟化（Datacenter Virtualization）：有的云操作系统 厂商将云作为数据中心的虚拟化扩展，他们将云操作系统实现为 一种基础设施和虚拟化资源的自动化管理工具，其产品形态是云 操作系统软件产品，典型的代表是VMware的vCloud
  - 基础设施供应（Infrastructure Provision）：像AWS和阿里云这样 的厂商则把云操作系统实现为一种基于其自有的基础设施资源对 外提供服务的软件系统，侧重于根据用户的需求灵活、按需、自 助化地配置虚拟化的资源，其产品形态主要是云服务

OpenStack开源云OS

- CloudStack是一个基于Java开发的、架构简单的、具有高可用性及 扩展性的云计算平台。目前CloudStack支持管理大部分主流的 hypervisors，如 KVM，XenServer，VMware，Oracle VM，Xen等

- CloudStack物理架构，从小到大分为：cluster、pod、zone、region

OpenStack的组件及其交互关系

- 计算：**Nova**、ZUN、QINLING
- 硬件管理：IRONIC、CYBORG
- 存储：**SWIFT**、CINDER、MANILA
- 网络：**NEUTRON**、OCTAVIA、DESIGNATE
- 共享服务：**KEYSTONE**、**PLACEMENT**、**GLANCE**、BARBICAN、KARBOR、SEARCHLIGHT
- 编排：HEAT、SENLIN、MISTRAL、ZAQAR、BLAZAR、AODH
- 负载集成：MAGNUM、SAHARA、TROVE
- 应用生命周期：MASAKARI、MURANO、SOLUM、FREEZER
- 门户：**HORIZON**

云计算发展的趋势

- 虚拟化技术的细分和发展：不仅仅是虚拟机和容器
  - 嵌入式虚拟化
  - Unikernel及其实现机制

- 不断深化、扩展的云：FAAS、多云计算、边缘计算、异彩纷呈的云

你是否有了一张“云图”