微服务与云计算

Importance 🌟

1. 从计算方式的演进，从单机计算到集群计算，再到云计算。请问集群计算和云计算有何联系和差异？云计算要解决的核心问题是什么？解决思路是什么？云计算中的核心概念是什么？

   - 集群计算与云计算的联系：

     计算方式的演进，从单机计算到集群计算再到云计算，云计算是在集群计算上进一步深化的结果。

     - 集群计算在单机计算的基础上，把资源的单机多进程共享扩展到跨计算机多级共享
     - 云计算则在集群计算的基础上，通过集约效应和规模效应把共享成本摊薄，做到服务可靠同时控制价格

   - 集群计算与云计算的区别：
     - 集群计算主要解决这些关键问题：资源的高可用性、网络的高吞吐性、服务的可扩展性
     - 云计算主要则解决这些关键问题：部署的弹性、实惠性，以及服务的可靠性
   - 云计算要解决的核心问题：
     - 程序要能到处运行
     - 要能到处成功运行
     - 同一机器上运行的程序相互隔离
   - 解决思路：虚拟化和虚拟机
   - 核心概念：资源共享、规模效应、集约效应

---

2. 要使应用程序在云上**到处**能运行，面临什么问题？要使应用程序在云上能到处**成功**运行，又面临什么问题？请分别举例说明。解决上述两个层面的问题，采取的策略是用虚拟机来运行应用程序，于是虚拟机有两个层级，请举例说明。第一个层次的虚拟机有哪些？请举两个例子。第二个层级的虚拟机有哪些？也请举两个例子。这两个层级的虚拟机解决问题的思路和策略分别是什么？

- 要能到处运行面临的问题：计算机与操作系统的型号不匹配。应用程序在运行之前通常编译成二进制的可执行文件，这种文件计算机型号以及操作系统型号高度绑定。而云平台由大量不同的计算机组成，同样的应用程序在不同型号的计算机或操作系统下将无法运行。

  例子：Windows 32位的程序不能直接在Windows 64位或者Linux 64位的操作系统上运行

- 要到处成功运行面临的问题：共享资源时的隔离性不足问题。在云环境中，为了提高资源利用率，一台物理计算机上通常会运行来自不同客户的多个应用程序。如果缺乏有效的隔离机制，这些程序在共享资源时会互相干扰，导致运行失败。

  例子：如果程序A和程序B都试图监听同一个网络端口，或者争用同一个文件/设备，就会发生冲突，导致其中一个程序无法成功运行。

- 虚拟机有哪两个层级，分别举两个例子：
  - 面向程序语言的虚拟机：例如JVM、Node.js
  - 面向运行环境构建的虚拟机：例如VMWare、Container
- 这两个层级的虚拟机解决问题的思路和策略：
  - 第一层，面向程序语言的虚拟机：开发者将源代码编译成与平台无关的中间代码（例如Java字节码），而不是机器代码。然后在目标机器上安装一个虚拟机软件（JVM），由它负责在运行时将中间代码解释执行或实时编译（JIT）为本地代码。这样程序只需要面对统一的虚拟机标准，从而屏蔽底层硬件和操作系统的差异。
  - 第二层，面向运行环境构建的虚拟机：将物理资源转化为逻辑资源，为每个应用构建独立的运行环境，为每个应用程序提供虚与实的映射功能。

---

3. 中小微企业将原有的业务信息系统迁移至云上，交由云服务商来运维，具有哪些好处？当前绝大部分大型和中型企业并未将业务信息迁移到云上，为什么？小微企业为什么愿意将其业务系统迁移上云？
   - 业务信息迁移上云的好处：
     - 免除基础设施建设与维护负担：企业不需要考虑机房建设、网络铺设等问题；
     - 降低人力成本：将系统运维工作外包给云服务商，企业不需要雇佣额外的IT运维人员；

    - 更好应对业务波动：云服务器具有弹性与可伸缩性，企业可按需租用资源，在业务高峰期动态增大资源请求，避免资源浪费；
- 为什么大型和中型企业并未将业务信息迁移到云上：
    - 数据安全与商业机密顾虑：云计算的本质是资源共享，将核心业务数据放在云上，意味着数据存储在云服务商的物理设备上。对于大中型企业，数据是其核心资产和商业机密，他们担心数据泄漏或者被篡改。
    - 自身负担得起：大中型企业通常已经拥有完善的自建机房和IT团队，迁移上云可能意味着对已有投资的浪费
- 为什么小微企业愿意将其业务系统迁移上云：
    - 正如前面介绍的迁移上云的好处，本身小微企业无力承担自建机房和雇佣运维的成本，迁移上云可以让小微企业以极低的成本获得先进的信息化能力，物美价廉。

---

4. 云计算中，服务方提供的资源包括计算、存储、网络、软件等，从软件角度看，云计算中有哪三种服务提供方式，请分别举例说明。
    a. IaaS：基础设施即服务（Infrastructure as a Service）
        - 服务商将计算（CPU）、存储（硬盘）等硬件资源进行虚拟化，以“虚拟机”的形式租赁给用户。用户获得的是一台安装了操作系统的空白计算机，用户拥有最高权限
        - 例子：Amazon EC2，阿里云ECS
    b. PaaS：平台即服务（Platform as a Service）
        - 服务商除了提供底层硬件，还提供了软件的运行环境（Runtime）和开发工具（SDK）。用户只需要将开发好的代码上传到云平台，平台会自动负责运行和运维。
        - 例子：Nde.js
    c. SaaS：软件即服务（Software as a Service）
        - 服务商直接向用户提供完整的应用程序功能，用户直接通过网络（通常是浏览器）使用软件
        - 例子：Microsoft Office 365，Gmail等

---

5. 中心化服务系统有什么不足？去中心化服务是什么含义？去中心化服务有哪三个明显的新特质？请举出三个非常著名的去中心化服务的应用系统？Paxox协议中的

共识达成中涉及proposer和acceptor的3趟交互，这三趟交互的具体内容是什么？画出在无故障时，共识达成中proposer和acceprot的状态机。

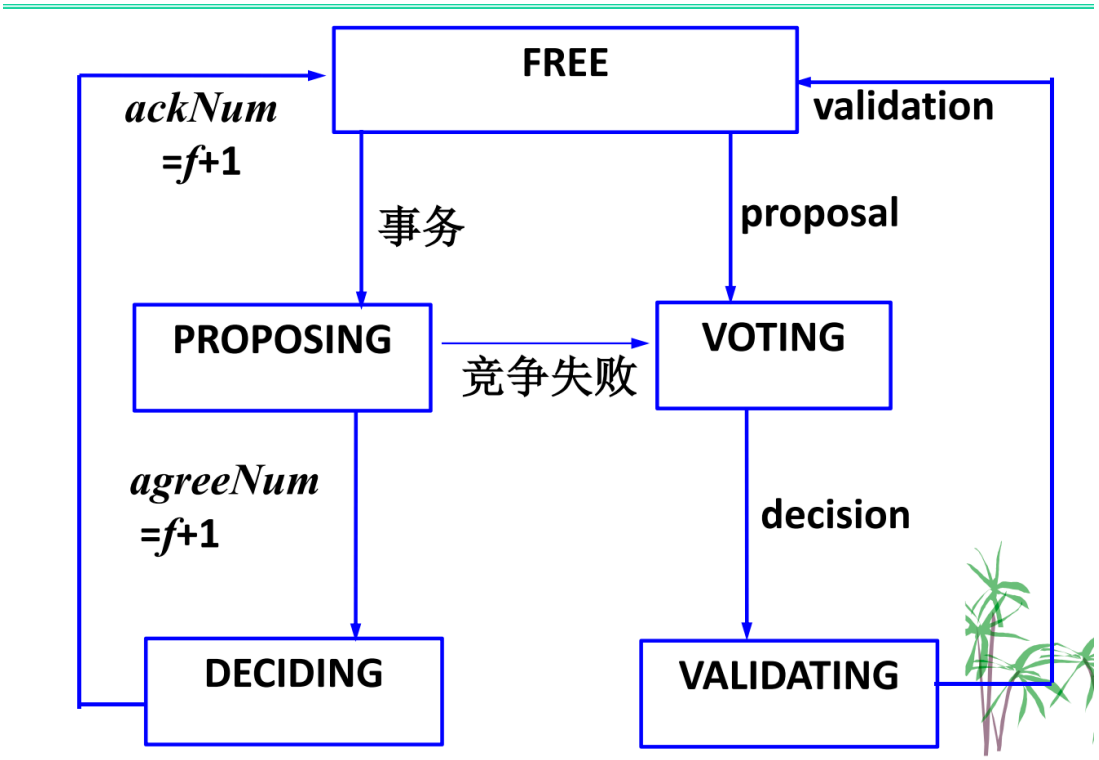
Paxos算法

- 中心化服务系统有什么不足：
  - 单点控制：服务由单一提供方掌控，数据的正确性、服务的公平性都以客户信任服务提供方作为前提
  - 数据不可控：数据的所有操作权限由服务方全权负责，客户无法掌控
  - 故障风险高：服务由单点提供，一旦出现单点故障，服务对客户不可用
- 去中心化服务的新特质：
  - 任何事物不再由单一的服务提供方说了算，而是由多个服务提供方共同商议决定
  - 在正常副本超过半数的前提下，任何一个副本对于服务的提供而言不再是必不可少，而是可有可无
  - 系统中即使由恶意副本想破坏系统一致性，都不会得逞
- 三个著名的去中心化应用系统：
  - 比特币
  - 以太坊
  - 树图区块链
- 共识达成涉及Proposer和Acceptor的三趟交互，这三趟交互的具体内容：
  1. 第一趟：Proposal
     - Proposer收到一个事务请求时，要为其申请一个全局唯一ID。如果Proposer在FREE状态，便广播一个proposal消息；要求送达超过半数的副本才算成功，随后进入PROPOSING状态
     - Acceptor在FREE状态收到一个proposal消息时，就把自己这张赞成票投给申请者；随后从FREE状态进入VOTING状态
  2. 第二趟：Decision
     - 如果Proposer在PROPOSING状态收到了超过半数的赞成票，便广播一个decision消息，同样要求送达超过半数的副本，随后进入DECIDING状态

- Acceptor在VOTING状态一旦收到decision消息，便给Proposer发送一个ack消息，随后进入VALIDATING状态

3. 第三趟：Validation
    - 如果Proposer在DECIDING状态收到超过半数的ack消息，广播一个validation消息，回到FREE状态
    - Acceptor在VALIDATING状态一旦收到validation消息，便将golbalQueue中对应事务的validation属性修改为TRUE，表示至少超过半数的副本同意该事务，共识达成可以执行该事务，最后回到FREE状态

- 无故障时共识达成的状态机



6. 服务器在云上运行有什么特点？对服务器在云上运行有什么要求？微服务的含义是什么？服务程序在哪几种情形下需要改版升级或更新？有哪些措施和手段来实现快速服务程序的更新？更换和切换分别是什么含义？哪一种好？这种好需要有什么前提条件？
    - 在云上运行有什么特点：
        - 动态性：服务在哪一台计算机上运行不能实现确定，要到运行时试试确定
        - 副本数也具有动态变化性

    - 运行可能要从一台计算机迁移至另外一台计算机
- 在云上运行有什么要求：
    - 启动要快，时间不能太长
    - 要解决在同一台机器上多个程序的资源（例如端口号）冲突和相互干扰问题
- 微服务的含义是什么？
    - 能到处运行、启动时间短、能正常运行的服务程序，就叫微服务程序
- 服务程序在哪些情形下改版升级：
    - 当程序有bug时
    - 当程序结构不合理时
    - 当功能要扩展时
    - 当功能实现要改进时
- 实现服务程序快速更新的手段：
    - 标准化：采用流行框架开发，代码命名和格式遵循标准，便于快速理解和修改
    - 自动化：尽量使用工具来自动化完成各环节的工作
    - 容器化/镜像化：将程序打包成镜像，运行在虚拟机中或者容器中，方便在每一个后续环节中可以开箱即用
- 更换和切换的含义：
    - 更换：关停并卸载老版本服务程序，然后安装和启动新版本服务程序。在更换期间服务不可用；
    - 切换：在另一台计算机上启动新版本服务程序，然后通过服务网关把客户请求从老版本切换至新版本。切换完毕后在关停老版本服务程序
- 哪一种好？有什么前提条件？
    - 显然切换更好
    - 前提条件：
        - 云计算环境：必须要能够快速获取另一台计算机来运行新版本程序
        - 虚拟化技术：支持快速创建和启动虚拟机/容器

---

7. 企业客户将其业务信息系统不加修改地迁移上云，涉及7步操作，这7步操作具体内容是什么？

   a. 注册企业账号并使用账号登录到云上

   b. 在云上创建虚拟系统，创建对应的局域网和虚拟机

   c. 给虚拟机设置所需的资源量

   d. 基于虚拟机的依赖关系设置虚拟机的启动顺序

   e. 使用Docker之类的软件工具将物理机上的文件系统打包成镜像文件

   f. 把镜像文件上传给云上对应的虚拟机

   g. 在云上启动企业信息系统

---

8. 云服务管理系统要启动企业客户的业务信息系统在云上运行，也涉及7步操作，这七步操作具体内容是什么？

   a. 确定由哪一台主机来运行虚拟机，给主机的Agent下达创建和配置虚拟机的命令

   b. Agent初始化虚拟机的`ipAddrMapTable`成员变量

   c. 初始化虚拟机的`volumeMapTable`成员变量

   d. 在主机上为虚拟机创建root目录，初始化虚拟机的`moutedDir`成员变量；下载镜像文件并解包至root目录

   e. 为虚拟机创建一个Namespace和Group，给虚拟机分配硬件资源以实现主机上虚拟机之间的隔离

   f. 启动虚拟机，创建一个子进程，将其与创建的VM、Namespace、CGroup内核对象关联

   g. 虚拟机捕获应用程序对bind函数的调用，初始化虚拟局域网上每个虚拟机的portMapTable成员变量，完成服务器侦听端口的虚实映射

---

9. 并行计算中的超算、集群计算、分布式计算，三者有何异同？

   a. 超算用来做超大规模科学计算。计算节点间数据交互频繁，处理器密集度高。使用专有网络通信。是一台计算机，昂贵、故障多，散热是突出问题。

   b. 集群用于数据中心，做超大规模数据处理。集群由同构的计算机组合而成，使用局域网连成一个大计算机系统。成员之间交互时可直接进行内存拷贝。与超算相比性价比高、运维简单、成本低、可扩展性好

c. 分布式系统是由自治和异构成员构成的邦联形式，即一种松散型的协同系统。成员间采用Web国际标准进行交互，交互时需要做转化翻译。和集群相比成员之间的交互具有低频性

---

10. 互联网应用包含前段和后端，面临的安全威胁可归纳为四类，具体是哪四类？分别简述应对这四类安全威胁的技术措施？

    - 四类安全威胁：
        a. 偷看：指未经授权获取信息
        b. 篡改：指未经授权修改信息
        c. 假冒：伪装成他人或合法服务
        d. 抵赖：用户否认自己曾经执行过某项操作
    - 应对措施
        - 应对偷看和篡改：使用对称加密技术对数据进行加密
        - 应对假冒：通过身份认证验证用户，通过HTTPS协议防止假冒网站
        - 应对抵赖：采用非堆成加密技术

---

11. 分布式数据库使用两阶段提交协议（2PC）来实现事务的四个属性，2PC协议包括哪三个子协议？这三个子协议分别处理什么问题？2PC协议有什么特性？

    - 提交协议：处理在无软硬件故障情形下，分布式系统中各成员如何协同来提交事务，以实现事务的原子性、持久性、一致性。
    - 终止协议：处理在有成员发生故障时，余下的正常成员如何应对
    - 恢复协议：有成员发生故障时如何恢复故障

    协议特性：绝大部分情况下终止协议时非阻塞的。只有在协调者和参与者同时故障，且正常成员全部投Ready票，但是都未收到协调者的commit/abort决定时，才会陷入阻塞状态

---

12. 函数调用是最简单的交互方式。函数调用有同进程内的函数调用，同机不同进程间的函数调用，以及跨机器边界的函数调用三种形式。前一种函数调用与后两种函数调用有什么不同之处？第三种形式又可分为同构机器之间的函数调用和异构机器之间的函数调用，它们之间又有什么差异？如何实现RPC对程序员透明？也就是说RPC的实现框架是什么？如何通过IDE来自动化得出有关RPC的源代码？

    - 同进程内的函数调用与后两种有什么不同：

    - 调用者和被调用者在同一个进程的地址空间中，参数传递可以通过寄存器或栈直接传输
    - 后两种函数的调用者和被调用者在不同的进程或物理机器，无法直接访问对方内存空间，需要通过操作系统提供的通信机制来完成交互
- 同构机器与异构机器之间函数调用的差异：
    - 同构机器之间：传输数据可以直接借助内存拷贝，不需要进行复杂的数据格式转换
    - 异构机器之间：数据交互时必须做翻译处理
- RPC的实现框架：
    - 客户程序：发出函数调用请求
    - 代理：位于客户端，将要调用的接口名与参数打包成请求包，发送给守护进程，然后等待结果返回
    - 守护进程：监听客户请求，加载服务模块，并调用存根
    - 存根：位于服务端，解析收到的请求包得到要调用的接口和实参，然后调用服务程序的接口，把调用结果返回给守护进程

  通过这种架构除用户调用外的逻辑都被封装，从而实现对程序员透明
- 如何通过IDE自动化得到RPC的源代码：

  基于接口，编写一个头文件，在其中定义服务函数的原型，随后IDE中的特定工具会扫描这个接口定义文件从而自动生成代理和存根的源代码。

---

13. 索引时提高数据定位查找的有效途径，它的原理是什么？哪种特性的数据适合使用树索引？哪种特性的数据适合使用哈希索引？哪种特性的数据适合使用混合索引，请分别举一个例子说明，给出索引方案，解释所给方案的合理性。
    - 原理：通过压缩将大块的数据映射成很小的索引，再通过对索引数据进行排序，降低查找复杂度
    - 树索引：
        - 适合的数据特性：需要范围查询、排序、分组、连接运算的数据
        - 例子：学生表中包含学号和姓名字段，可以基于这两个字段创建索引；因为树索引保持了数据的有序性，当需要按学号或姓名顺序查找学生时，可沿着树结构快速定位
    - 哈希索引：

    - 适合的数据特性：等值查询，大数据量下的精确匹配
    - 例子：选课表，包含stu_id, course_info等字段。可以以stu_id创建哈希索引，因为每个学生只需要查看自己的选课信息
- 混合索引：
    - 适合的数据特性：访问频次极高且具有明显层次结构的数据
    - 例子：电商订单，首先根据用户ID创建哈希索引，定位具体用户的所有订单；再通过订单日期创建树形索引，用于范围查询或分页

---

14. 交换机、路由器、将局域网接入公网的网关，这三个设备有何共性，有什么不同？交换器和网桥又有什么不同？要求从软件视角和网络协议视角来回答这几个问题。
    - 共性：都在做“存储-查表-转发”的工作
    - 不同：
        - 从协议视角：交换机工作在数据链路层，路由器工作在网络层，网关则同时在网络层和传输层中工作
        - 从软件视角：交换机关注MAC地址，路由器关注IP地址，网关关注IP+端口

    交换器和网桥的不同：

    - 从软件角度：网桥维护一个哈希表或红黑树用于查找，交换机则使用TCAM查找
    - 从协议视角：网桥的转发模式是存储转发必须完整接受一个帧，交换机是直通转发，只读到目标MAC就开始转发

---

15. 以面向对象编程的观念来看，网络对象（network）和网络连接（connection）对象，他们是不是有从属关系，谁从属于谁？是不是一对多关系？分别用哪些成员变量标识？在接受IP数据包时，network对象如何知晓它属于那个connection对象？当发送一个数据包时，connection对象如何知晓它属于哪个network对象？
    - 从属关系：Connection从属于NetWork，一个Network对应多个Connection
    - 标识：Network用本机IpAddr标识网络对象；connection用本端port标识连接

- 接收时：遍历connectionList，用目标端口匹配对应的connection
- 发送时：connection根据内部的linkage成员变量，这个变量指向了所属的Network