分布式系统背景

集中式系统：集中式系统中整个项目就是一个独立的应用，整个应用也就是整个项目，所有的东西都在一个应用里面。

集中式系统具有明显的优点，开发测试运维会比较方便，不用为考虑复杂的分布式环境。

但其弊端就是不易扩展，每次更新都必须更新所有的应用，而且，一个有问题意味着所有的应用都有问题。当系统越来越大，集中式将是系统最大的瓶颈。

分布式系统：分布式系统是若干独立计算机的集合，这计算机对用户来说就像单个相关系统。也就是说在一个分布式系统中，一组独立的计算机展现给用户的是一个统一的整体，用户感知不到背后的逻辑，就像访问单个计算机一样。

在分布式系统中，应用可以按业务类型拆分成多个应用，再按结构分成接口层、服务层；我们也可以按访问入口分，如移动端、PC 端等定义不同的接口应用；数据库可以按业务类型拆分成多个实例，还可以对单表进行分库分表；增加分布式缓存、搜索、文件、消息队列、非关系型数据库等中间件……

分布式系统可以解决集中式不便扩展的弊端，我们可以很方便的在任何一个环节扩展应用，就算一个应用出现问题也不会影响到别的应用。

分布式存储系统简介

分布式存储系统，是将数据分散存储在多台独立的设备上。

与分布式系统的基本构思一致，对于传统的网络存储系统存在的问题：

存储服务器成为系统性能的瓶颈，可靠性和安全性不能满足，无法应对大规模存储应用……分布式网络存储系统采用可扩展的系统结构，使用多台存储服务器分担存储负荷，位置服务器定位存储信息，提高了系统的可靠性、可用性和存取效率，且易于扩展。

分布式存储系统类型分类

块存储和文件存储是我们比较熟悉的两种主流的存储类型，而对象存储是一种新的网络存储架构。

块级是指以扇区为基础，一个或我连续的扇区组成一个块，也叫物理块。它是在文件系统与块设备（例如：磁盘驱动器）之间。

文件级是指文件系统，单个文件可能由于一个或多个逻辑块组成，且逻辑块之间是不连续分布。

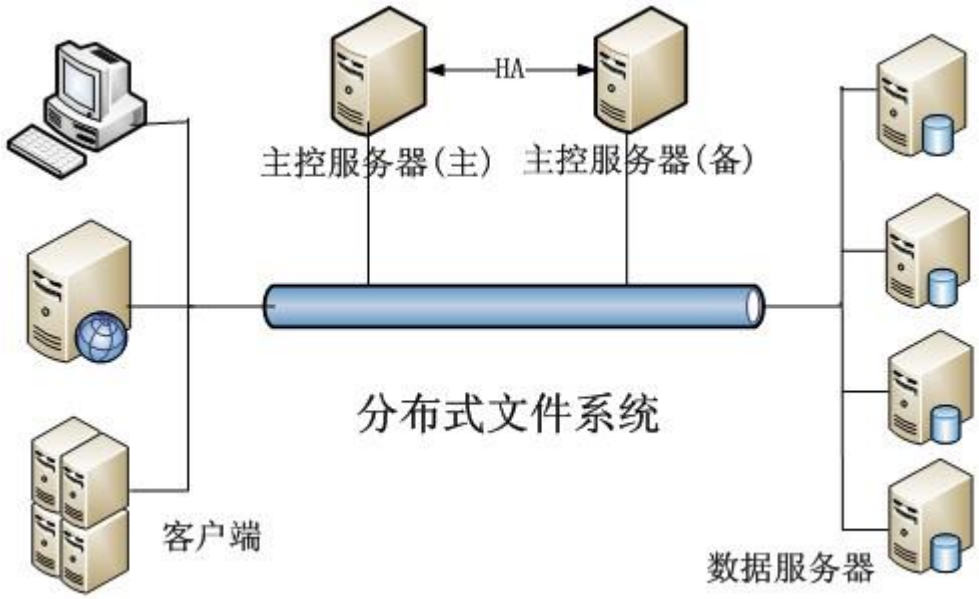
映射关系：扇区→物理块→逻辑块→文件系统

对象是系统中数据存储的基本单位，一个对象实际上就是文件的数据和一组属性信息的组合，这些属性信息可以定义基于文件的 RAID 参数、数据分布和服务质量等，而传统的存储系统中用文件或块作为基本的存储单位，在块存储系统中还需要始终追踪系统中每个块的属性，对象通过与存储系统通信维护自己的属性。在存储设备中，所有对象都有一个对象标识，通过对象标识 OSD 命令访问该对象。通常有多种类型的对象，存储设备上的根对象标识存储设备和该设备的各种属性，组对象是存储设备上共享资源管理策略的对象集合等。

分布式存储系统典型架构

目前比较主流的一种分布式文件系统架构，如下图所示，通常包括主

控服务器（或称元数据服务器、名字服务器等，通常会配置备用主控服务器以便在故障时接管服务，也可以两个都为主的模式），多个数据服务器（或称存储服务器，存储节点等），以及多个客户端，客户端可以是各种应用服务器，也可以是终端用户。



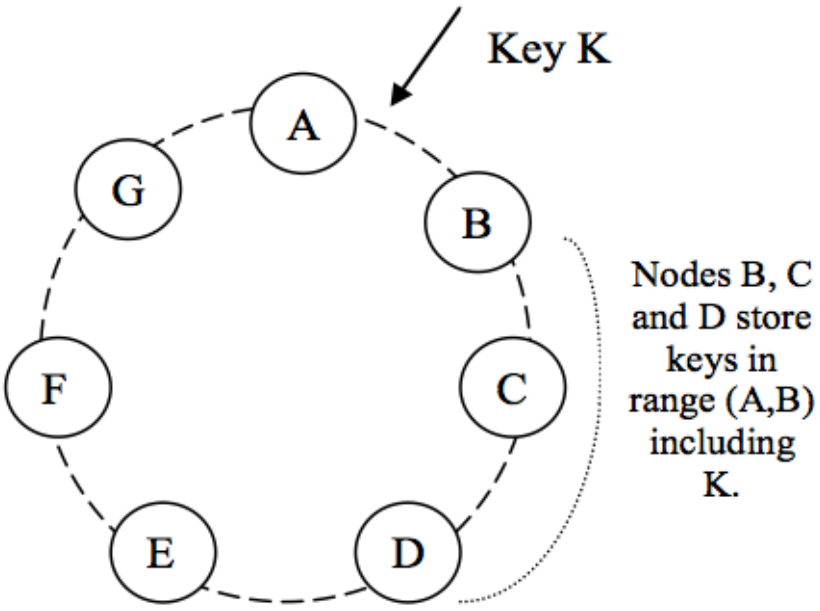
分布式文件系统的数据存储解决方案，归根结底是将将大问题划分为小问题。大量的文件，均匀分布到多个数据服务器上后，每个数据服务器存储的文件数量就少了，另外通过使用大文件存储多个小文件的方式，总能把单个数据服务器上存储的文件数降到单机能解决的规模；对于很大的文件，将大文件划分成多个相对较小的片段，存储在多个数据服务器上。

分布式存储系统算法

1.HASH :一个简单直观的想法是直接用 Hash 来计算，简单的以 Key 做哈希后 对节点数取模。可以看出，在 key 足够分散的情况下，均匀性 可以获得，但一旦有节点加入 或 退出 时，所有的原有节点都

会受到影响。稳定性无从谈起。

2.一致性 HASH：一致性 Hash 可以很好的解决 稳定性问题，可以将所有的 存储节点 排列在收尾相接的 Hash 环上，每个 key 在计算 Hash 后会顺时针找到先遇到的 存储节点 存放。而当有节点 加入或退出时，仅影响该节点在 Hash 环上 顺时针相邻 的 后续节点。但这有带来均匀性的问题，即使可以将存储节点等距排列，也会在 存储节点个数变化时带来数据的不均匀。



3. 带负载上限的一致性 HASH：一致性 Hash 有 节点变化时不均匀的问题。Google 在 2017 年提出了 Consistent Hashing with Bounded Loads 来控制这种 不均匀的程度。简单的说，该算法给 Hash 环上的每个节点一个 负载上限 为 1 + e 倍的 平均负载，这个 e 可以自定义。当 key 在 Hash 环上 顺时针 找到合适的节点后，

会判断这个节点的 负载 是否已经 到达上限，如果 已达上限，则需要继续找 之后的节点 进行分配。

4. 带虚拟节点的一致性 HASH：为了解决 负载不均匀 和 异构 的问题，可以在 一致性 Hash 的基础上引入 虚拟节点。即 hash 环上的每个节点 并不是 实际 的 存储节点，而是一个 虚拟节点。实际的存储节点 根据其 不同的权重，对应 一个 或 多个虚拟节点，所有落到相应虚拟节点上的 key 都由该 存储节点负责。

5.分片：分片将哈希环切割为相同大小的分片，然后将这些 分片 交给不同的节点负责。一个 节点退出 时，其所负责的 分片 并不需要顺时针合并 给之后节点，而是可以更灵活的 将整个分片 作为一个整体 交给 任意节点。在实践中，一个 分片 多作为 最小的数据迁移 和 备份单位。

6.CRUSH 算法：CRUSH 算法本质上也是一种 基于分片 的数据分布方式，其试图在以下几个方面进行优化：

分片映射信息量：避免中心目录服务和存储节点 及 客户端之间 交互大量的 分片映射信息，而改由 存储节点 或 客户端 自己根据 少量 且 稳定 的集群节点拓扑和确定的规则自己计算分片映射。

完善的故障域划分：支持 层级 的 故障域控制，将 同一分片 的 不同副本 按照配置划分到 不同层级 的 故障域中。

客户端 或 存储节点 利用 key、存储节点 的 拓扑结构 和 分配算法，独立的进行 分片位置 的计算，得到一组负责对应 分片 及 副本 的 存储位置。