discussion02

1354202

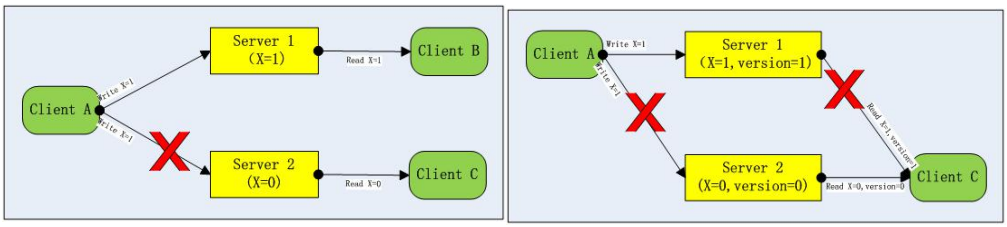
1 分布式：高可用性，高吞吐量

问题：

可用性和一致性（所有节点拥有数据的最新版本）的矛盾。分区容忍--现实：容忍网络出现分区，分区之间网络不可达。

分析：

虽然无法达到同时达到强一致性和极致可用性，但我们可以根据数据类型在二者中选择其一后去优化另外一个，Paxos 协议就是一种在保证强一致性前提下把可用性优化到极限的算法。



方案：

CAP 理论：在分布式环境下网络分区无法避免，需要去权衡选择数据的一致性和可用性，Paxos 协议提出了一种极其简单的算法在保障数据一致性时最大限度的优化了可用性，Zookeeper 的 ZAB 协议把 Paxos 更加简化，并提供全局时序保证，使得 Paxos 能够广泛应用到工业场景。

2 数据存储，分区，一致性，缓存

问题：

数据一致性问题，分区可用性问题，性能问题。

分析：

缓存：

1 本地缓存:数据存储在应用代码所在内存空间.优点是可以提供快速的数据访问;缺点是数据无法分布式共享,无容错处理.典型的,如 Cache4j;

2 分布式缓存系统:数据在固定数目的集群节点间分布存储.优点是缓存容量可扩展(静态扩展);缺点是扩展过程中需要大量配置,无容错机制.

3 弹性缓存平台:数据在集群节点间分布存储,基于冗余机制实现高可用性.优点是可动态扩展,具有容错能力;缺点是复制备份会对系统性能造成一定影响.典型的,如 Windows Appfabric Caching。

4 弹性应用平台:弹性应用平台代表了云环境下分布式缓存系统未来的发展方向.简单地讲,弹性应用平台是弹性缓存与代码执行的组合体,将业务逻辑代码转移到数据所在节点执行,可以极大地降低数据传输开销,提升系统性能.典型的,如 GigaSpaces XAP.

方案：

数据一致性，通过日志的强同步，可以解决。

Cassandra 通过 4 个技术来维护数据的最终一致性，分别为逆熵（Anti-Entropy），读修复（Read Repair），提示移交（Hinted Handoff）和分布式删除。

分区可用性，在出现任何异常情况时仍旧保证系统的持续可用，可以在数据强同步的基础上引入 Paxos/Raft 等分布式一致性协议来解决，虽然这个目前没有成熟的实现。

ebay 使用的 BASE 思想（(basically available, soft state, eventually consistent）。

Cassandra 中，Token 是用来分区数据的关键。分区策略的不同，Token 的类型和设置原则也有所不同。 Cassandra (0.6 版本)本身支持三种分区策略。

性能：对于单个事务来说，RT 增加。其响应延时一定会增加（至少多一个网络 RT，多一次磁盘 Sync）；对整个数据库系统来说，吞吐量不变。远程的网络 RT 和磁盘 Sync 并不会消耗本地的 CPU 资源，本地 CPU 的开销并未增大。只要是异步化做得好，整个系统的吞吐量，并不会由于引入强同步而降低。

多级缓存：客户端页面缓存（http header 中包含 Expires/Cache of Control，last modified(304，server 不返回 body，客户端可以继续用 cache，减少流量)，ETag）；反向代理缓存；应用端的缓存(memcache)；内存数据库 Buffer、cache 机制（数据库，中间件等）。

3 负载均衡

问题：

选择哪种负载，需要综合考虑各种因素（是否满足高并发高性能，Session 保持如何解决，负载均衡的算法如何，支持压缩，缓存的内存消耗）。

分析：

一个大型的平台包括很多个业务域，不同的业务域有不同的集群，可以用 DNS 做域名解析的分发或轮询，DNS 方式实现简单，但是因存在 cache 而缺乏灵活性；

一般基于商用的硬件 F5、NetScaler 或者开源的软负载 lvs 在 4 层做分发，当然会采用做冗余(比如 lvs+keepalived)的考虑，采取主备方式。

4 层分发到业务集群上后，会经过 web 服务器如 nginx 或者 HAProxy 在 7 层做负载均衡或者反向代理分发到集群中的应用节点。

方案：

LVS，工作在 4 层，Linux 实现的高性能高并发、可伸缩性、可靠的的负载均衡器，支持多种转发方式(NAT、DR、IP Tunneling)，其中 DR 模式支持通过广域网进行负载均衡。支持双机热备(Keepalived 或者 Heartbeat)。对网络环境的依赖性比较高。

Nginx 工作在 7 层，事件驱动的、异步非阻塞的架构、支持多进程的高并发的负载均衡器/反向代理软件。可以针对域名、目录结构、正则规则针对 http 做一些分流。通过端口检测到服务器内部的故障，比如根据服务器处理网页返回的状态码、超时等等，并且会把返回错误的请求重新提交到另一个节点，不过其中缺点就是不支持 url 来检测。对于 session sticky，可以基于 ip hash 的算法来实现，通过基于 cookie 的扩展 nginx-sticky-module 支持 session sticky。

HAProxy 支持 4 层和 7 层做负载均衡，支持 session 的会话保持，cookie 的引导；支持后端 url 方式的检测；负载均衡的算法比较丰富，有 RR、权重等。
对于图片，需要有单独的域名，独立或者分布式的图片服务器或者如 mogileFS，可以图片服务器之上加 varnish 做图片缓存。

4 系统监控

问题：

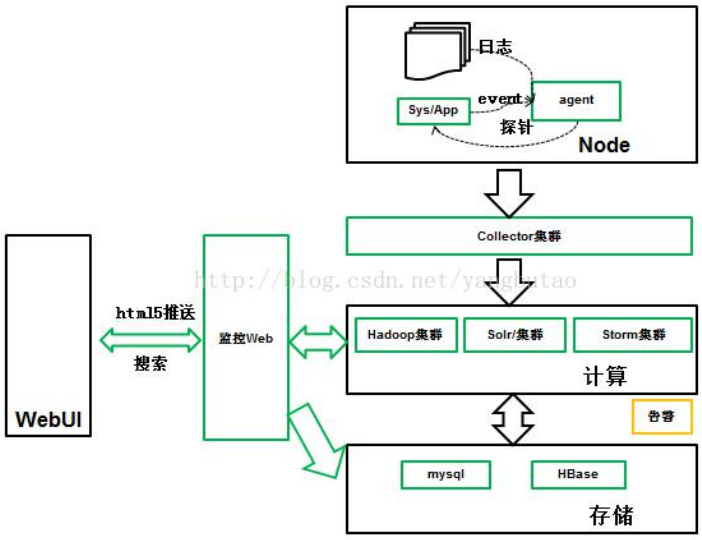
监控平台的性能、吞吐量、已经可用性就比较重要，需要规划统一的一体化的监控平台对系统进行各个层次的监控。

分析：

平台的数据分类：应用业务级别：应用事件、业务日志、审计日志、请求日志、异常、请求业务 metrics、性能度量；系统级别：CPU、内存、网络、IO

时效性要求：阀值，告警；实时计算；近实时分钟计算；按小时、天的离线分析；实时查询；

方案：



5 通信可靠高效

方案：

JGroups 框架。

6 消息队列

问题：

应用耦合，异步消息，流量削锋等问题。实现高性能，高可用，可伸缩和最终一致性。

分析：

在实际应用中常用的使用场景。异步处理，应用解耦，流量削锋和消息通讯四个场景。

方案：

一般商用的容器，比如 WebLogic，JBoss，都支持 JMS 标准，开发上很方便。但免费的比如 Tomcat，Jetty 等则需要使用第三方的消息中间件。常用的消息中间件（Active MQ,Rabbit MQ，Zero MQ,Kafka）。

7 协议

问题：

多个节点间数据同步问题。

方案：

简单有效，totem 协议；paxos 协议；gossip 协议。

8 安全

问题：

单站点故障、网络故障等自然因素引起的；本机或网络上的人为攻击。
海量实时数据流和多元化用例场景下数据行为监控的需求。

方案：

安全措施根据如下几点：访问控制、安全隔离、数据分类、数据加密以及实时数据行为监控。
实时分布式 Hadoop 数据安全方案 - Apache Eagle，旨在提供高效分布式的流式策略引擎，并集成机器学习对用户行为建立 Profile 以实时智能地保护 Hadoop 生态系统中大数据安全。