Problem 2

1352978 施闻轩

需求

使用传统的多线程模型开发的网络应用有 C10K 问题，通过使用合理线程池+异步套接字（本项目所采用的技术）可以提高单节点的服务能力，达到 C10M。因此若需要进一步提高服务能力，需要设计分布式架构。

具体问题

1. 基础服务

对于分布式服务来说，首先需要解决基础设施问题，包括多机备份、自动热迁移等。这可以使用商业化的云服务简化部署，例如可以使用 Google Cloud Computing, Amazon AWS, Microsoft Windows Azure, Aliyun 等云计算服务。这些云计算服务自建了可靠的数据多机和异地备份、热迁移等特性，并具有非常高的 SLA。在这些云计算服务之上，可以进一步搭建分布式的架构。

2. 性能

负载均衡

由于存在多个服务节点，因此需要处理用户网络请求的分配。一般有以下几种方法解决这个问题：

- 在客户端内置一批 IP 列表或 DNS 列表：连接服务端时从中随机挑选，或根据时间由算法挑选，或由用户挑选一个，达到负载均衡。
- 采用基于时间变化 IP 或基于地域变化 IP 的动态 DNS 服务实现负载均衡（其中，DNS 服务的架构不在讨论之列）：使用这种方法时，客户端不需要硬编码 IP 地址，可以应对 IP 地址变化的情况。
- 在入口处采用多个四层或七层前端负载均衡器：使用这种方法时，由于可以使用算法分配用户请求到一个压力较小的服务器上，因此负载均衡效果比以上两种更稳定。
- 对于 UDP 服务可以使用 IP Anycast 技术实现地域级别的负载均衡。

以上方法可以组合使用，例如首先根据地域划分出 Asia 节点域名，America 节点域名，Europe 节点域名等，而每个地域域名可以进一步解析到多个 IP 地址上，每个 IP 地址都运行着硬件级负载均衡器集群，负责将请求再分发到集群中的一个计算节点进行处理。

缓存

在分布式环境下，缓存也需要是分布式的。常见的缓存方案如 Redis, Memcached 都具有集群功能，可以很好地在分布式环境下工作，是理想的选择。对于该项目来说，由于不存在数据库，也没有很耗时的操作，因此不需要缓存。

3. 可靠性和健壮性

Unique ID Generator

在分布式环境下，由单机生成 ID 的方案会产生单点瓶颈，因此需要有分布式的唯一标示符生成器。

- UUID (GUID): 时间和空间上确保唯一性，时间上的唯一性通过时间参数来实现，空间上的唯一性是通过 MAC 地址和随机数等参数实现。
- Twitter Snowflake: 比 GUID 简单、高效，通过为不同机器指定不同的 Machine ID 参数确保同一时间不同空间上的唯一性。

其他分布式唯一标示符生成器大多数都基于以上两种变化而来，例如有些 Snowflake 的变种算法使用 MAC 地址而不是参数来指定 Machine ID 字段，实现零配置。

消息队列

通过使用消息队列，可以在分布式环境下进行各个组件的解耦合，提高鲁棒性。常见的消息队列如 ActiveMQ，RabbitMQ 和 NSQ 均为分布式环境设计，可以很好地在分布式环境下工作，并支持非常多的编程语言。

4. 一致性

在分布式环境下，确保节点间数据的一致性是一个很关键的问题。有以下协议可以同步集群节点状态：

- 两阶段递交、三阶段递交、paxos 协议
- gossip 协议

5. 安全性

时间服务

大部分算法正常工作都有一个假设是时间是一致的、同步的，因此时间服务是确保整体系统安全的重要基石。一般使用 NTP 协议确保节点间时间同步。

签名

使用非对称加密算法可以对消息进行签名和验证。

- RSA
- ECC

其中，ECC 在相等密钥大小情况下有更强的安全性，性能上更优，但 ECC 的安全性还取决于所选择的曲线参数，这些曲线有潜在的被间谍机构植入后门的风险，相比之下，RSA 更可信。

加密

一般使用对称加密算法对消息进行加密，它们有足够强的安全性，又比非对称加密有更高的性能。

- RC4 (Weak)
- DES
- AES