何清

173-7354-6038
heqing2023@email.szu.edu.cn
https://blog.csdn.net/fantasticHQ
https://github.com/UtopianYouth

Education

深圳大学计算机与软件学院，计算机技术（硕士） 2023.09 – 2026.07

湖南工业大学计算机学院，计算机科学与技术（本科） 2019.09 – 2023.07

技术栈

- 熟悉 Linu 系统编程，如进程管理（fork/exec）、IPC 机制、POSIX 线程库等，具备多线程同步开发能力；
- 熟悉 Linux 网络编程，I/O 多路复用机制，能够基于 Reactor/Proactor 模式构建高并发网络服务；
- 熟悉 OSI 七层网络模型、TCP/IP 协议栈等；
- 熟悉数据结构及常见算法，如数组、链表、栈、队列、二叉树、查找与排序、BFS 和 DFS 等；
- 熟悉 MySQL 基本操作，表的约束、范式、索引，具备数据库设计能力；
- 熟悉 MVVM 软件架构模式，具备使用 Cpp 和 QT5 实现业务逻辑与 UI 层高效解耦的应用开发能力；
- 了解 xv6 操作系统源码，通过修改内核代码实现进程调度、IPC 和信号量机制，实验记录；
- 熟悉 Git 等开发工具，软件设计师（中级）。

实习经历

深圳市智物联网络有限公司 2025.07 – 2025.09

后端开发工程师

- **负责 API 分组与信息管理模块的开发测试：**实现 API 分组管理和 API 信息管理 Model 层、logic 层及 API 层；优化 API 分组管理的多叉树生成算法，处理父子节点关系；使用 Apipost 完成接口的功能性测试，为 yunuis 平台提供清晰的 API 层级管理能力。
- **研究与实现微服务架构下的 HTTP 请求转发：**分析 consul 服务注册与发现机制，实现从 consul 动态获取所有注册微服务及其 action 列表到 yunuis 常驻内存；使其接收请求后精准路由至对应微服务执行业务逻辑。解决 yunuis 解析 system action 调用本地 plus 包下的 API，无法动态发现和调用外部微服务提供 action 的问题，实现 yunuis 支持微服务调用的请求透明转发。
- **设计并实现多类型 Action 的请求分发与执行机制：**基于已有的工作内容一和二，通过哈希表缓存 Action 列表；针对 Internal Action 利用 Go 的反射机制，查找并执行对应结构体方法；System Action 复用已开发的微服务 HTTP 转发器进行路由；Custom Action 集成 mixgo 框架的 Lua 脚本引擎执行用户自定义逻辑。为 yunuis 构建统一、高效且可扩展的请求处理管道，使 yunuis 能够分发和执行三种不同来源的业务逻辑。

项目经历

一、轻量级 Linux 多线程 Web 服务器

项目源代码

- **项目描述：**该项目基于 Cpp 开发在 Linux 下的轻量级多线程 Web 服务器，使用线程池、IO 多路复用、有限状态机、线程同步等技术，实现处理 HTTP 请求；通过 EPOLL 事件通知机制和设置 fd 非阻塞的伪异步 IO 模拟 Proactor 网络 I/O 模型，测试其性能达到 10k 到 15k QPS。
- **线程池：**解决在高并发场景下，频繁创建工作线程处理 HTTP 请求的低效率问题；
- **IO 多路复用：**通过 EPOLL 事件通知和设置 fd 非阻塞，实现 TCP 读写缓冲区非阻塞 I/O，提高服务器并发效率；
- **有限状态机：**通过状态转移机制，高效解析 HTTP 请求头、请求行和请求体；
- **定时器**：将 TCP 连接客户端信息封装成定时器类，定时器对象用双向链表存储，主线程收到 SIGALRM 信号触发定时器逻辑，定期检测非活跃的 TCP 连接；
- **线程同步**：对 Linux 下的互斥锁和信号量封装，工作线程互斥访问工作队列；
- **内存映射和分散写**：将 HTTP 请求的资源从磁盘映射到用户内存区，主线程调用 writev() 分散写，HTTP 响应头、状态行和响应体内容从用户内存区拷贝到内核 TCP 写缓冲区。

二、分布式实时消息推送系统

项目源代码

- **项目描述：**分布式实时消息推送系统通过三级解耦架构，实现 comet 层，job 层和 logic 层，使用 Kafka 和 gRPC 实现服务间通信；本系统将连接管理、业务逻辑与消息推送分离，确保了其高可用性和水平扩展能力，为需要高并发、低延迟消息推送的场景提供稳定可靠的后端服务；本系统支持房间隔离消息、全局广播等多种消息模式，提供完整的用户认证、消息持久化与历史消息查询等功能。
- **comet 层：高效管理 WebSocket 长连接，并将消息实时、可靠地推送到客户端**
  - **基于 muduo 网络库的 Reactor 模型：**通过非阻塞 I/O 和 Reactor 事件驱动模式，高效处理数万并发连接；
  - **连接与会话管理：**通过 ConnectionManager 管理所有客户端 websocket 连接，使用 PubSubService 管理房间与用户的订阅关系，实现高效地广播消息推送；
  - **gRPC 服务端：**将 gRPC 接口给 Job 层调用，解析 job 层的 Protobuf 消息，通过订阅管理机制找到对应连接进行消息推送。
- **job 层：从 kafka 获取推送消息，调用 comet 层消息推送逻辑，实现业务逻辑与消息推送解耦**
  - **kafka 消费者：**使用 librdkafka 高效，从 kafka 指定 Topic 中消费消息，kafka 的顺序性、持久化和至少一次投递特性，保证消息不丢失；
  - **消息路由与转换：**解析 Protobuf 消息 (PushMsg)，根据 type 和 roomId 字段，路由到正确的推送逻辑；
  - **gRPC 客户端：**维护与 comet 层的 gRPC 连接池，根据消息类型，调用不同的 gRPC 方法，将消息分发到一个或多个 Comet 服务器上。
- **logic 层：处理所有 HTTP 请求、实现业务逻辑和操作存储层，如用户注册登录、消息存储、房间管理等**
  - **RESTful HTTP API：**基于 Muduo 实现的 HTTP 服务器，提供清晰的 JSON API 接口，便于前端调用和调试；
  - **数据持久化：**使用 MySQL 进行数据的持久化存储（用户信息、房间信息、全量消息记录）。使用 Redis 缓存用户 session 和热点数据；
  - **kafka 生产者：**消息广播业务逻辑完成，构造 Protobuf 格式推送消息，通过 librdkafka 异步生产到 Kafka 中，通知 Job 层进行推送。