Bai Zixuan

+86-156-5070-5080 bzx0619@gmail.com github.com/napallday

Basic Information

- Nationality: Chinese, **Singapore Permanent Resident** (No visa sponsorship required)

Education

Bachelor, Electronics Engineering, Peking University(北京大学 信息科学技术学院电子学系) | *Beijing, China*

- GPA: 3.6/4.0 (Rank:10/50); **Dual Degree: Economics** **Sep 2014 - July 2018**

Master, Signal & Information Processing, Peking University | *Beijing, China* **Sep 2018 - April 2021**

(voluntarily quit master program because it's not related to computer science)

Awards & Publications

- 2019 IEEE Student Competition **First Prize**
- Publications: ICASSP, INFOCOM, ICC, IEEE IoT Journal, IEEE Network

Working Experience

Senior Software Engineer, Marketplace Tech Services, Shopee *Singapore* **May 2021 - June 2024(3 years)**

- Responsible for building reliable and sustainable technical products and providing solid solutions in middleware areas to empower the technical robustness and sustainability for Marketplace product lines and backend teams
- **Promoted to Senior Software Engineer in one year**
- **Projects**
    - **In-house Golang Kafka Client Library**

      Built an in-house unified Golang Kafka client library in Shopee, used by **20+ teams** in **2000+ projects**. The design initiated from experiences and best practices, making Kafka interactions performant, safer and easier. I also gave a company-wide technical sharing on Kafka

        - Supported Tracing, Metrics, Full Chain Stress Test, Message Persistence, Traffic Routing(forward, double write, filter), Config Hot Reload
        - Built an **out-of-the-box consumer model** -- With this model, it only takes several minutes to build up a performant and reliable consumer service from scratch

          Core features includes:

            - Concurrent Message Transformation
            - Message Processing in different concurrent levels(by partition, by key, by any field in biz model, randomly)
            - Hot Retry, Multi-layer Cold Retry, Dead Letter Queue
            - Message Batch Processing
            - Advanced Offset Management to avoid message loss
        - Provided Kafka SPI/SP in a company-wide service development framework(like SpringBoot)
        - Avoided common pitfalls(infinite rebalance), integrated best practices(new partitions automatic detection) and optimizations(best-effort message de-duplication), etc
    - **High Performant Message Transformation/Cache Invalidation Platform**

      A high performant & scalable solution to monitor Database events for cache invalidation or message format transformation

        - Built a light-weight and concurrent ETL pipeline framework, adopted in 1000+ projects
        - **Cache Invalidation Platform**: A centralized place where developers can effortlessly manage their cache invalidation process

        - Read from Kafka binlog messages to invalidate cache
        - Adopted delay double deletion policy to avoid DB slave delay issue
    - **Message Transformation**: A general solution that transforms database binlog to customized data model by Kafka
        - Features include: Data Filtering, Field Mapping, Field Deletion, Field Generation, Triggering Conditions, Debounce, Re-partition, Split by Region, etc

Intern, Ads Infra, ByteDance *Beijing* **April 2020 - September 2020 (6 months)**

Open Source Contributions And Personal Projects

IBM Sarama - Golang Client for Apache Kafka with over 11k stars

- Implemented **incremental cooperative rebalance protocol**
    - Rewrote core consumer logic including setup/cleanup, rebalance, offset management and heartbeat
    - **Eliminated STW(Stop-The-World) problem** during rebalances, reduced rebalance overhead by 90+% and made a company-wide tech sharing
- Resolved the long-standing issue of **automatic partition detection**, which is the main reason Alibaba Cloud's recommend not using Sarama
- Implemented **multiple rebalance strategies support**, enabling zero-downtime rolling upgrades
- Fixed various performance issues including **race conditions**, **CPU burning**, and **memory optimization**

Collateralized Lending Protocol - Solidity DeFi Project

A lending protocol allowing users to mint synthetic stablecoins against their collateral

- Implemented **floating bonus mechanism** to incentivize liquidators and enhance protocol robustness
- Applied **CEI(Check-Effect-Interaction) pattern** and added reentrancy guard to avoid reentrancy attacks
- Integrated with Chainlink Price Feeds and added **stale price protection**
- Applied several **gas optimization** tricks
- Used **static analysis** tools(slither, aderyn) to find potential issues
- Developed comprehensive test suite with mock tests and **handler-based invariant(stateful) fuzzing** using Foundry

Huff Counter - Low-level Smart Contract Implementation

- Implemented the same counter contract in four different abstraction levels: **Solidity, Solidity & Yul, pure Yul, and Huff**
- Applied **differential fuzzing tests** to ensure behavioral consistency across implementations

Skills

- **Programming Languages**: Golang, Java, Solidity
- **Web2**: Kafka, Redis, MySQL
- **Web3**: Solidity, Foundry, EVM(opcodes)
    - Familar with Common Attack Vectors, Gas Optimization, Common ERC protocols, Uniswap, Compound, etc
- **Strengths**: Eager to learn new things, Problem Solving, Communication