Xie Yuchen

+65 87906385 | xieyuschen@gmail.com | linkedin.com/in/xieyuschen | github.com/xieyuschen |

Education

Huazhong University of Science and Technology (HUST) Wuhan, China
Bachelor of Engineering in Information Security *Sep. 2018 – June 2022*

Experience

Shopee Singapore
Senior Software Engineer, Marketplace Technical Service *Jan. 2025 – Present*

Project: Dynamic Go GC Tuner

- Architected adaptive GC tuning framework for 76 services (116k instances, 680K CPU cores, 1.1PB memory) during 12.12 campaign with zero incidents; saved **$0.6M** (17K CPU cores). Projected CPU savings with 95% accuracy pre-rollout by modeling Go's GC pacing formula and fitting historical live heap/GOGC metrics from Grafana.
- Executed gradual 50% rollout per SDU with multi-layer guardrails (live heap auto-reset at 50% threshold, 85% memory ceiling); created SOP and alert thresholds, communicated memory usage expectations (65–75% range) to PICs to address concerns proactively.
- Leveraged runtime/metrics API to dynamically tune GOGC via `debug.SetGCPercent` and `SetFinalizer`; reduced GC cycles by 85% (48.3 to 7.7 per 2min) and GC CPU cost by 82% while maintaining P99 latency SLO.

Project: Combined Compile and Deployment (CD)

- Engineered a custom build toolchain to compile disparate microservices into a single binary, replacing network RPCs with direct in-process function calls. Eliminated serialization overhead and achieved production-ready deployment across 6 services with zero incidents.
- Bypassed Go's Minimal Version Selection (MVS) by rewriting import paths pre-compilation, allowing logically independent services to run conflicting dependency versions within the same memory space.
- Achieved logical namespace separation for system resources by duplicating and patching the Go standard `os` package. Virtualized `os.Getenv` with service-specific prefixes and rerouted `os.Stdout` to prevent cross-service pollution.
- Integrated the unified binary workflow with existing deployment and observability platforms. Engineered context propagation for in-process calls to guarantee accurate trace ID passing, log routing, and metrics tagging without relying on network interceptors.

Software Engineer, Marketplace Technical Service *Aug. 2022 – Dec. 2024*

- **Go-Application-Server (GAS):** Contributed to the GAS ecosystem to streamline library version upgrades across large-scale services; currently adopted by all Shopee marketplace services. Core contributor to config and toolchain that supports library/Go version control.
- **Xproto, HTTP-Spex Transprotocol Support:** Designed the end-to-end workflow for HTTP clients and servers to migrate from the internal SPEX RPC system (including HTTP router extraction, Protobuf parsing and code generation, and HTTP URL-to-RPC command conversion).
- **Uniconfig:** Developed a configuration SDK enabling fast, concurrent access to configurations from various data sources.

Projects

gopls-mcp & Go GitHub Member | *Go, gopls, MCP Server* May 2025 – Present

- **Go GitHub Member:** Contributor to `golang/tools` and `golang/go`, improving `gopls`—the official Go language server used by millions of developers.
- **gopls-mcp:** Identified the AI-LSP gap, forked gopls to build an MCP wrapper, exposing its core semantic analysis capabilities to AI agents to enable precise code navigation beyond plain text search.

Technical Skills

Languages: Go (expert), SQL, Bash
AI & Developer Tools: Toolchain, LSP/gopls, AI Code Agent workflow
Infrastructure: Git, CI/CD, tracing/monitoring/logging, RPC, Protobuf