Firstname Lastname

City, State (Open to relocation) • you@email.com • github.com/yourhandle • linkedin.com/in/yourhandle

SUMMARY

Senior Software Engineer specializing in Rust and systems programming. Proven record delivering low-latency, memory-safe services at scale, leading migrations from C++/Go to Rust, and elevating reliability through rigorous testing, observability, and SRE practices.

CORE SKILLS

Languages: Rust (async, lifetimes, generics, unsafe/FFI), C/C++, Go, Python
Systems: Linux, networking, filesystems, IPC, multithreading, performance tuning
Rust Ecosystem: Cargo, Tokio, async-std, serde, tonic/gRPC, prost, anyhow/thiserror, tracing, criterion, proptest, quickcheck, libFuzzer/AFL, Clippy, Miri
Tooling: Bazel, Git, CI/CD (Buildkite/GitHub Actions), Docker, Kubernetes, OpenTelemetry, Prometheus, Grafana
Domains: Distributed systems, microservices, RPC, streaming, storage, security/hardening

EXPERIENCE

Senior Software Engineer (Rust) • **Google** • www.google.com • *City, State* MM YYYY – Present

- Led migration of a performance-critical service from C++ to Rust, cutting p99 latency by 35% and reducing memory-related incidents to zero over 6 months.
- Designed and shipped a Rust-based, async RPC proxy (Tokio + tonic) handling >150k RPS with p99 <10 ms under peak load; improved CPU efficiency by 25% via zero-copy data paths.
- Introduced property-based and fuzz testing pipelines (proptest, libFuzzer) uncovering >30 defects pre-production; defined reliability SLOs and on-call practices reducing incident rate by 40%.
- Built observability with OpenTelemetry tracing and Prometheus metrics; reduced mean time to detect by 60%.
- Authored internal Rust guidelines, mentored 8 engineers on ownership/concurrency patterns, and conducted >150 code reviews.
- Implemented safe FFI bridges to existing C++ libraries; replaced unsafe blocks with audited abstractions to meet security review requirements.

Software Engineer (Rust/C++/Go) • **Previous Company** • www.previouscompany.com • *City, State* MM YYYY – MM YYYY

- Implemented high-throughput, fault-tolerant services; introduced async Rust components where latency and safety were critical.
- Drove performance profiling (perf/eBPF) and workload benchmarking to guide capacity planning and cost reductions.

PROJECTS

High-Throughput Stream Processor (Rust)

- Async pipeline using Tokio; backpressure-aware design; achieved 2x throughput vs. prior Go service with stable p99 latency.

Secure Config Service (Rust)

- Implemented end-to-end validation, schema evolution, and client-side caching with exponential backoff; zero production regressions over 12 months.

OPEN SOURCE & COMMUNITY

Contributor to Rust ecosystem crates (e.g., serde/tracing-adjacent tooling). Presented internal tech talks on async Rust and safe FFI patterns.

EDUCATION

B.S. or M.S., Computer Science (or related field) • **University Name** • *City, State* • YYYY.

CERTIFICATIONS

(Linux Performance, Cloud/Kubernetes, or Security credentials if applicable)

KEYWORDS FOR ATS

Rust, Tokio, async, ownership/borrowing, lifetimes, FFI, C/C++, gRPC, Protobuf, distributed systems, observability, OpenTelemetry, Prometheus, fuzzing, property-based testing, Bazel, Cargo, Linux, performance profiling, SRE, reliability.