

Evgeniy Kramsakov

Senior Rust Engineer · Saint Petersburg, Russia · remote or hybrid

t.me/sobieg sobiegrus@gmail.com github.com/geibos

ABOUT

Rust engineer with a background in applied cryptography. I build network and backend systems, from eBPF in the Linux kernel to services on Tokio and Kafka.

I like building whole systems, from the kernel to operations. I develop with Claude Code and Codex and validate changes with tests, load-testing setups and measurements.

EXPERIENCE

Aug 2024 –
present

Senior Rust Engineer · MTS Web Services (MWS)

- Interview candidates, onboard new engineers and review code. Own the architecture of my service and contribute to the architecture of the whole service system.
- Develop an eBPF/XDP traffic filter: kernel and userspace in Rust, SNI-based decisions from TLS ClientHello, events to Kafka.
- Founded the MWS Rust guild with about 100 members: meetups and engineering practice discussions.
- Built a TRex load-testing setup and found three scenarios in which the filter silently let traffic through.
- Report export: from 21 to 8 s and from 849 to 61 MB of memory. NetFlow collector: fixed a memory leak, peak down from 863 MB to under 100 MB.
- Ported the filter to AArch64, added TCP RST metrics and simplified its state machine.
- Migrated two Java services to Rust with AI agents: migration plan, ADRs and an invariant registry first, then compatibility tests (268 unit and 38 contract tests for the subscriber access service), then the implementation. Deviations from the original are logged separately.
- Researching a methodology for comparing Go and Rust in agent-driven development on SWE-bench Multilingual tasks (in progress).
- Maintained a filtering DNS server on a hickory-dns fork: DoT, blocking statistics and reasons, preparation for production use.
- Wrote a filter snapshot versioning service from scratch on axum and SeaORM: tests, criterion benchmarks, migrations.
- Worked on the anti-tracking statistics service (Kafka → PostgreSQL → S3): Kafka source, export to S3. Brought OpenSpec to the team.
- First task: a Rust bridge between UCI and a USP agent on OpenWrt routers: FFI, ubus, cross-builds for aarch64 and mipsel.
- Built an admin panel in Rust and axum for a neighbouring product's business feature settings.
- Team: two Rust engineers, a Go tech lead, a Go engineer, a systems analyst and a product manager.

Rust · Tokio · eBPF/XDP · aya · axum · Kafka · PostgreSQL · S3 · Go

Jan 2023 – Jul
2024

Rust Engineer · Scalable Solutions

- Developed the Rust core of a 2-of-3 threshold signature scheme for a product for cryptocurrency exchanges: side-channel protection, execution inside an HSM.
- Built a library for running code inside a hardware security module (HSM).
- Covered the code with unit, integration and end-to-end tests and wrote the documentation.
- Team: me and the lead in cryptography, 5 Scala engineers, QA and DevOps around us.

Rust · threshold signatures · HSM · constant-time

Dec 2021 – Jan
2023

Cryptography Researcher · Demlabs

- Researched post-quantum schemes for a blockchain and implemented lattice-based ring signatures in C.
- Designed a staking algorithm; implemented lattice-based linkable ring signatures and RingCT and reviewed the security of my implementation.
- Introduced static analysis and covered the C code with GoogleTest unit tests.
- Cryptanalysis and prototypes in Python and SageMath. The only researcher on a team of four or five engineers.

C · post-quantum cryptography · lattices · CMake · GoogleTest · Python

Apr 2020 – Dec
2021
Saint Petersburg

Software Engineer · Practical Security Systems

- Implemented the GOST Magma cipher in C for OpenSSL for secure data exchange between EAEU countries and evaluated it for side-channel leakage as part of certification.
- Integrated a hardware random number generator.
- Leakage evaluation method: running under DOS to remove operating system noise, checking that memory is cleared.
- Cut generation of a large volume of derived keys from 18 hours to 2 without deviating from the specification.
- Restructured key derivation based on how symmetric ciphers work: fast computational algorithms and bit tricks.
- Worked with Qt and OpenSSL. A team of five or six.

C · C++ · Qt · OpenSSL · GOST · Linux

SKILLS

Languages	Rust, C, Python; also Go, C++, TypeScript
Networking	Linux, eBPF/XDP (aya), AF_XDP, TLS, DNS, NetFlow, OpenWrt, load-test rigs with TRex
Backend	Tokio, axum, PostgreSQL (SeaORM, sqlx), Kafka, Redis, S3, gRPC, OpenTelemetry
Cryptography	GOST, post-quantum, threshold and ring signatures, side channels, HSM
Infrastructure	Docker, GitLab CI, ArgoCD, Helm, Prometheus, Grafana, Traefik, Authentik
AI agents	Claude Code, Codex, OpenSpec: specification → compatibility tests → implementation → review
Testing	unit, integration, contract, end-to-end and load tests (TRex and VM-based setups); proptest, criterion benchmarks, GoogleTest

PROJECTS

Home server

About forty services behind Traefik with single sign-on and monitoring. An LLM gateway on LiteLLM observed through Langfuse, shared memory for AI agents across machines.

ozi-rs

An offline raster map and track editor for search and rescue teams: Rust, Tauri 2, Svelte 5, MapLibre.

la-proxy

A wrapper around telemt, a Telegram proxy: self-service through a bot, personal links, device limits, administration. Rust, SQLite, Docker.

agent-board

A self-hostable mirror of getpostingboard.dev, a shared board for AI agents. TypeScript.

Core War engine

An ICWS'94 assembler and simulator in Rust, checked against the reference pMARS.

check_inet

Internet connection diagnostics in a single binary: shows which layer is broken.

Planning Poker

Story point estimation without sign-up: real-time rooms, anonymous voting. React, TypeScript, socket.io. The tool was incorporated into a corporate task tracker.

audio-test

Speaker layout check for a multichannel audio system in the browser via the Web Audio API.

lipflow-rs

A Rust port of silent dictation by lip reading, faster than the Python original. Stopped: Russian recognition is not accurate enough.

PacketSender

A university L2–L4 packet generator with a Python and Qt interface: batch sending and load scenarios, up to taking down a home router.

root-only-file-access-LSM

A Linux security module: files in a directory are accessible to root only.

VOLUNTEERING

May 2025 – present

IT Lead, Search Group Leader · LizaAlert, Saint Petersburg

- Reviving the team's IT division since September 2026, search group leader since July 2026.
- Path in the team: searcher since May 2025 → operations cartographer since April 2026 → search group leader since July 2026 → IT lead since September 2026.
- Running an inventory of home-grown IT solutions across about twenty divisions: who built what, what is used, where the risks and personal data are.
- Training with the special operations group.

2025

Active Volunteer award · MTS

- Take part in the company's volunteer initiatives.

EDUCATION

2015 – 2022

Peter the Great St. Petersburg Polytechnic University, Institute of Computer Science and Cybersecurity · Specialist degree, Computer Security

- Thesis: the post-quantum isogeny-based SIDH protocol for a drone swarm. Reproduced the Castryck–Decru attack on it in July 2022.
- Major in Mathematical Methods of Information Security.
- SIDH is a post-quantum key agreement protocol based on elliptic curve isogenies.
- A link between a base drone and a swarm of small microcontroller-based vehicles.
- Elliptic curve isogenies were chosen for their small keys, which matters for microcontroller memory.
- Cryptography, penetration testing, reverse engineering, database security, physical security and side-channel leak prevention, microcontrollers.
- Extended the Pintos teaching OS (Stanford) to multithreading and wrote an application for it.

UNIVERSITY ACTIVITIES

Chair of the dormitory student council

A council of 10–15 people. Events of up to a hundred participants and small gatherings of 10–15.

Organising committee of a city photo exhibition

Negotiations with sponsors and organisers.

First-year student mentor

Helped first-year students settle into university and dormitory life.

LANGUAGES

Russian

native

English

B1–B2: read and write technical documentation, little speaking practice