Pratham K

prathamIN@proton.me | blog | @git-bruh

Technical Skills

Tools: Git, Docker, GDB, Strace, GNU Make, CMake, Meson, Autotools
Languages: Bash / POSIX sh, C, C++, Python, Rust, TypeScript, Zig
Miscellaneous: CI/CD, Self Hosting, Linux Kernel Configuration, Software Packaging, Systems Administration

Professional Experience

System Engineer Aug 2023 – Present

Subconscious Compute

- Integrated code coverage visualization and performance reports in CI using **Rust** language tooling like **grcov**, **flamegraph**, and system profiling tools like **perf** to catch performace regressions and facilitate code coverage
- Developed Dockerfiles and Bash/Powershell scripts for CI pipelines to execute tests, and cross-compile release packages to various targets on Windows, Linux (**deb** and **rpm** formats) and MacOS platforms
- Implemented observability into system events on MacOS using the **Endpoint Security** library to generate alerts and execute user-specified actions in their response
- Led the end-to-end development of various backend services adhering to third-party protocol specifications, including an Apple **MDM** server to remotely manage devices and enforce policies

Open Source Experience

Package Maintainer and Core Team Member 2022 – Present

KISS Linux Community

- Participated in the packaging and upkeep of software packages including Compiler Toolchains, Containerization Tools and Browsers, working with build systems like **CMake** and **Meson**, and writing **runit**-based service scripts
- Developed automation projects to streamline the maintenance workflow, including the implementation of a sandboxed multi-stage rootfs bootstrap script utilizing **unshare** and **bubblewrap**: maintainer-utils

Open Source Contributor 2022 – Present

Notable Contributions To Projects Used In Personal FOSS Endeavours

- Resolved a bug in **Chromium** that caused page crashes on **GCC** builds due to undefined behavior: #4546610
- Contributed new interfaces, portability fixes, and support for mouse events to **termbox2**, a TUI library: termbox2
- Implemented validation for nullable fields in the GraphQL schema in **tailcall** as part of a paid bounty: #521
- Fixed a crash in the **nouveau** driver's firmware loading code due to erroneous usage of the Linux DMA API: #24

Projects

Kaldi ASR Client | *C++, Python, CMake, OpenSSL, gRPC, NVIDIA Triton* Dec 2022 (Contract)

- Developed a C++ client library to perform audio inference on WAV files with **Kaldi** and **NVIDIA Triton Inference Server**, utilizing **gRPC** for communication
- Created Python bindings with **ctypes**, addressing signal and exception handling concerns in the library interface
- Created a build pipeline to build C++ libraries like **Kaldi** from source and integrate them with Python bindings into a Python wheel, addressing build system quirks with tools like **patchelf**

dabba.rs | *Rust, User Namespaces, CGroups, slirp4netns* Sep 2023 - Present

- Developed a small, fully rootless container runtime akin to **runc** using Kernel APIs like **User Namespace** and **CGroups**, with the ability to run basic **OCI**-complaint container images, using **OverlayFS** to mount layers
- Implemented networking between the container namespace and host using **TAP** devices via **slirp4netns**, allowing programs to connect to the internet from the container and expose ports to the host system

landbox | *C, Make, Linux Syscalls* Oct 2022 - Present

- Developed a CLI and helper library inspired by **bubblewrap**, using the Linux **Landlock** API for filesystem sandboxing allowing restriction of read, write and execute permissions for arbritary paths

Matrix TUI | *C, Meson, cJSON, libcurl, lmdb, termbox2* Aug 2021 - Present

- Developed a minimal TUI to interact with the Matrix communication protocol, effectively leveraging queues, pthreads, pipes, signals, and the poll() syscall to implement asynchronous logic
- Created libmatrix using **libcurl** and **cJSON** to wrap the Matrix REST APIs, offering clean API interfaces with patterns like tagged unions and iterators, and implemented an efficient key-value event store using **LMDB**
- Wrote unit tests, and integrated sanitizers, static analyzers and coverage reports to improve code quality & safety