Tucker J. Polomik

Email: tuckerpolomik@gmail.com
Phone: (845) 381-8007
GitHub: https://github.com/tuckerpo
LinkedIn: https://linkedin.com/in/tuckerpo
Personal Site: https://tuckerpo.me

Experience

- **CableLabs** — Louisville, CO
  Lead Software Architect — August 2024 - Present
  Senior Software Engineer — July 2022 - August 2024
  - Owned full-stack system architecture and execution, spanning hardware bring-up, kernel subsystems, user-space design, and interface layers; guiding technical direction while remaining deeply hands-on in implementation, code review, and problem-solving.
  - Made upstream contributions to open-source networking projects such as OpenWRT, prplOS, RDK-B, prplMesh, and hostapd.
  - Patched kernel drivers for Qualcomm IPQ40xx and IPQ80xx Wi-Fi SoCs (ath10k, ath11k)
  - Wrote a Python/JavaScript web UI for visualizing EasyMesh network topologies.
  - Wrote a Java/C++ application that interfaces to power meters for automated hardware-in-the-loop testing.
  - Worked on various networking innovation projects, resulting in several pending patents.
  - Developed Mobile WiFi, a novel technology that facilitates seamless WiFi client roaming between different physical access points. This innovation, adopted by cable industry leader Liberty Global, ensures uninterrupted WiFi connectivity while users are on the move.
  - Responsible for hiring, on-boarding, and mentoring interns and junior engineers.
  - Architected and implemented a system that allows users to configure and converse with their WiFi routers using natural language. (C, node.js, JavaScript, HTML, CSS)
  - Developed POSIX userspace process that collects PHY layer metadata about network traffic. Serves data via binary protocol over Unix/TCP sockets.
  - Represented CableLabs at conferences, talks, and technology demonstrations, acting as a technical communication channel for novel networking technologies.
- **QSC** — Boulder, CO
  Software Engineer — Nov 2021 - July 2022
  - Developed and maintained Q-SYS, a software platform for managing networked audio/video equipment.
  - Developed user interface components for multiple platforms and patched Linux device drivers against Rockchip RK356x aarch64 SoCs.
  - Developed asynchronous, multi-threaded SNMP library (1m requests/s, <10 ms latency).
  - Patched numerous memory leaks and hard faults that were causing customers deployed products to crash: AddressSanitizer (ASAN), gdb, valgrind.
  - Exposed C++ libraries to user-facing Lua scripting engine.
- **INFICON** — Syracuse, NY
  Software Engineering Lead — July 2020 - November 2021
  Software Engineer — September 2018 - July 2020
  - Led software engineers in the development of mass spectrometers, gas chromatographs, and thin-film deposition monitors (3 direct reports).
  - Worked closely with customers to translate market feedback into engineering solutions and served as a scrum master.
  - Developed complete embedded Linux systems for our products using Yocto: kernel, rootfs, device tree, device drivers, POSIX userspace applications.
  - Developed, extended, and backported Yocto BSPs for embedded x86 and ARM SoC deployment.
  - Developed Linux kernel drivers (subsystems: USB, network, TTY)
  - Developed Linux kernel driver and POSIX userspace hooks to emulate a 16 bit ISA DMA controller, replacing deprecated hardware with a software solution.
  - Developed custom protocol layer, standardized internally for a unified TCP API across our product line.
  - Developed cross platform library for bootloading Intel/Altera FPGAs from a host PC, or microcontroller.
  - Designed/developed firmware for embedded x86 and ARM deployments (I2C, USB2.0, RS232, RS485, SPI, watchdog timers).
  - Developed RS485 messaging front-end for Intel/Altera FPGAs (softcore NIOS II, VHDL).
  - Reverse engineered a competitor product using IDA, gdb, binutils, wireshark, tcpdump.
  - Developed novel time-series data peak detection algorithm.
  - Reverse engineered and extended SHARC DSP assembly.

Patents

- **Encrypted Beacons**: Method to encrypt IEEE802.11 Beacon Management Frames [Filed Non-Provisional]
- **WiFi On-Boarding Interception**: Method for on-boarding a single device to a BSS [Filed Non-Provisional]
- **Method for Detecting Path Obstructions Between 802.11 Access Points and Clients**: Estimate a WiFi access point's location and RF path using noise measurements. [Filed Provisional].

Publications

- I. Wheelock, S. Arendt, S. Glennon, **T. Polomik**, Z. Foreman: Virtualization in Wi-Fi to Fix Many Long-Standing Customer Experience Issues: National Cable & Telecommunications Association (NCTA), October '23.

Open Source Contributions gitlab.com/tpolomik github.com/tuckerpo github.com/tuckerpo-cl

- **RDK-B**: Major contributor to RDK-B's EasyMesh implementation, OneWifi, and hardware abstraction layer (HAL). C, C++, Embedded Linux, Yocto.
- **prplMesh**: Major contributor to the prpl Foundation's implementation of the Wi-Fi EasyMesh standard. Principally developed most of the Virtual BSS (VBSS) feature defined in R6 of the EasyMesh standard. C++
- **prplOS**: I contributed to the prpl Foundation's fork of OpenWRT, canonically named prplOS. My contributions included backporting of upstream OpenWRT packages, novel packages, and patches to both the Linux kernel as well as hostapd.
- **topologyViewer**: Python web application that enumerates and visualizes WiFi EasyMesh networks. Used extensively by the prpl Foundation to demonstrate prplMesh features. Additionally serves as a CPE/STA management platform, allowing users to graphically steer clients between different physical access points, remotely.
- **stationSniffer**: Linux userspace C++ application that sniffs Wi-Fi frames and coalesces different PHY layer metrics about a network, including RSSI, MCS index, and noise. Data is served via Unix or TCP sockets to clients who can leverage this metadata to make intelligent decisions about a network, such as steering a client device to a more suitable access point.
- **Linux Kernel Module Programming Guide (LKMPG)**: Audit and submit changes to the Linux Kernel Modules Programming Guide. Changes include fixing function and structure signatures, correcting typos, and adding additional information for Linux kernel subsystems I am familiar with and have developed for.

Programming & Tooling

- **Languages**: C, C++, Python, shell, Java, JavaScript, assembly, Go
- **Tools & Frameworks**: GNU tools, CMake, git, Yocto, Visual Studio, Qt, Docker
- **OS/Platforms**: Linux, Windows, OpenWRT, FreeRTOS, Bare-metal.
- **Hardware**: Comfortable reading schematics & using oscilloscopes, logic analyzers, multi-meters, etc.

Education

- **University at Buffalo, School of Engineering** — Buffalo, NY
  Bachelor of Science, Computer Engineering — Awarded September 2018
- **Linux Foundation** — Syracuse, NY
  Embedded Linux Platform Development with Yocto Project — Awarded December 2018
- **Linux Foundation** — Syracuse, NY
  Linux Foundation Certified Engineer - Linux Kernel Internals — Awarded December 2019