Contribution ID: **65** Type: **not specified**

“Is Upstream Really Enough?”—Practical Realities of Using eBPF in Long-Term Supported Systems

eBPF has emerged as a foundational technology for building observability, networking, and security tooling across modern Linux systems. However, in long-term supported (LTS) and embedded environments—such as automotive or industrial platforms—the deployment of eBPF-based software remains fraught with challenges. These range from kernel version divergence and verifier incompatibilities to inconsistent user-space toolchains and even organizational security policies that prohibit the use of bpf() altogether.

I will explore the real-world pain points of using eBPF in such constrained contexts. Key technical topics will include:

- Kernel internals depending on eBPF programs are not stable even if tracepoints
- Verifier Incompatibility
- libbpf / libclang hell: Toolchain inconsistencies and CI unfriendliness in embedded environments.

Beyond technical aspects, I will examine cultural and organizational mismatches—such as differing definitions of “backport”, lifecycle misalignments between kernels and eBPF tooling, and CI burden across multiple kernel versions.

As part of this discussion, I will introduce a proposed initiative: the “eBPF Embedded Profile” —a structured subset of APIs, helpers, kfuncs, and tooling practices aimed at enabling safe and predictable eBPF use in embedded or production-grade LTS environments. This profile would also include guidance for runtime stability, security vetting, and deployment feasibility and may serve as a stepping stone toward future standardization or conformance efforts.

Primary author: TADA, Kenta

Presenter: TADA, Kenta

Session Classification: eBPF Track

Track Classification: eBPF Track