android

eBPF on Android

IETF 116
March 2023

Why use eBPF?

- Full kernel performance and access without touching kernel
  - Run code in kernel context with no kernel changes (we **still** support kernel 4.9…)
  - Stable ABI: same code runs on all kernels
  - Access to in-kernel data via helper functions
  - Communication to userspace via eBPF maps
  - Safety guaranteed by in-kernel verifier

- Original motivation: replace xt_qtaguid, which was 3000 lines of out-of tree code (Android Pie)
  - Powers `TrafficStats` and `NetworkStatsManager` Android APIs

Current uses of eBPF

- Pretty much every networking packet hits eBPF
  - Data usage and accounting
  - Firewalling / network restrictions for power saving
  - High-speed packet processing
    - 464xlat, tethering (~4 Gbps USB tethering on Pixel 6!)
  - Network tracing via eBPF ringbuffers
  - TCP/UDP port reservations, DSCP remarking, …
- Tracepoints