Contribution ID: **138** Type: **not specified**

Scaling Linux Traffic Shaping with BPF

Thursday, November 15, 2018 9:10 AM (20 minutes)

Google servers classify, measure, and shape their outgoing traffic. The original implementation is based on Linux kernel traffic control (TC). As server platforms scale so does their network bandwidth and number of classified flows, exposing scalability limits in the TC system - specifically contention on the root qdisc lock.

Mechanisms like selective qdisc bypass, sharded qdisc hierarchies, and low-overhead prequeue ameliorate the contention up to a point. But they cannot fully resolve it. Recent changes to the Linux kernel make it possible to move classification, measurement, and packet mangling outside this critical section, potentially scaling to much higher rates while simultaneously shaping more flows and applying more flexible policies.

By moving classification and measurement to BPF at the new TC egress hook, servers avoid taking a lock millions of times per second. Running BPF programs at socket connect time with TCP_BPF converts overhead from per-packet to per-flow. The programmability of BPF also allows us to implement entirely new functions, such as runtime configurable congestion control, first-packet classification and socket-based QoS policies. It enables faster deployment cycles and as this business logic can be updated dynamically from a user agent. The discussion will focus on our experience converting an existing traffic shaping system to a solution based on BPF, and the issues we've encountered during testing and debugging.

Presenters: BRUIJN, Willem de (Google); HAO, Eddie (Google); DUMITRESCU, Vlad (Google)

Session Classification: BPF MC