Contribution ID: **165** Type: **not specified**

BPF control flow, supporting loops and other patterns

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

Currently, BPF can not support basic loops such as for, while, do/while, etc. Users work around this by forcing the compiler to "unroll" these control flow constructs in the LLVM backend. However, this only works up to a point. Unrolling increases instruction count and complexity on the verifier and further LLVM can not easily unroll all loops. The result is developers end up writing code that is unnatural, iterating until they find a version that LLVM will compile into a form the verifier backend will support.

We developed a verifier extension to detect bounded loops here,

https://git.kernel.org/pub/scm/linux/kernel/git/bpf/bpf-next.git/log/?h=wip/bpf-loop-detection

This requires building a DOM tree (computationally expensive) and then matching loop patterns to find loop invariants to verify loops terminate. In this discussion we would like to cover the pros and cons of this approach. As well as discuss another proposal to use explicit control flow instructions to simplify this task.

The goal of this discussion would be to come to a consensus on how to proceed to make progress on supporting bounded loops.

I agree to abide by the anti-harassment policy

Presenter: FASTABEND, John (Cilium)

Session Classification: BPF MC