Contribution ID: **323** Type: **not specified**

Using SCEV to establish pre and post-conditions over BPF code

Wednesday, September 11, 2019 5:20 PM (20 minutes)

Currently, the BPF verifier has to "execute" code at least once and then it can prune branches when it detects the state is the same. In this session we would like to cover a technique called Scalar Evolution (SCEV) which is used by LLVM and GCC to perform optimization passes such as identifying and promoting induction variables and do worst case trip analysis over loops. At its most basic usage SCEV finds the start value of variables, the variables stride and the variables ending value over a block of code. Building a SCEV pass into the BPF verifier would allow us to create a set of pre and post conditions over blocks of BPF codes.

We see this as potentially useful to avoid "executing" loops in the verifier and instead allowing the verifier to check pre-conditions before entering the loop. And additionally establishing pre and post conditions on function calls to avoid having to execute the verifier on functions repeatedly. We suspect this will likely be necessary to support shared libraries for example.

The goal of the session will be to do a brief introduction to SCEV. Provide a demonstration of some early prototype work that can build pre and post conditions over blocks of BPF code. Then discuss next steps for possible inclusion.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr FASTABEND, John (Isovalent)

Presenter: Mr FASTABEND, John (Isovalent)

Session Classification: BPF MC