Contribution ID: **122** Type: **not specified**

Detecting semantic bugs in the Linux kernel using differential fuzzing

Wednesday, 22 September 2021 07:10 (25 minutes)

Many bugs are easy to detect: they might cause assertions failures, crash our system, or cause other forms of undefined behaviour detectable by various dynamic analysis tools. However, certain classes of bugs, referred to as *semantic bugs*, cause none of these while still resulting in a misbehaving faulty system.

To find semantic bugs, one needs to establish a specification of the system's intended behaviour. Depending on the complexity of the system, creating and centralising such specifications can be difficult. For example, the *"specification"* of the Linux kernel is not found in one place, but is rather a collection of documentation, man pages, and the implied expectations of a vast collection of user space programs. As such, detecting semantic bugs in the Linux kernel is significantly harder than other classes of bugs. Indeed, many test suites are meant to detect regressions, but creating and maintaining test cases, as well as covering new features requires significant amounts of engineering effort.

Differential fuzzing is a way to automate detection of semantic bugs by providing the same input to different implementations of the same systems and then cross-comparing the resulting behaviour to determine whether it is identical. In case the systems disagree, at least one of them is assumed to be wrong.

syz-verifier is a differential fuzzing tool that cross-compares the execution of programs on different versions of the Linux kernel to detect semantic bugs. It was developed as part of the syzkaller project which also provides unsupervised coverage-guided kernel fuzzing.

To generate programs, syz-verifier uses a declarative system call description language called syzlang. This allows generating valid random programs (sequences of system calls) the same way as syzkaller does. The programs are then dispatched for execution on different versions of the Linux kernel. After programs finish executing, the produced results (currently only errnos returned by each system call) are collected and verified for mismatches. In case a mismatch is identified, syz-verifier reruns the program on all kernels to ensure it is not flaky (i.e. consistently reproducible rather than triggered due to some background activity or external state). If the mismatch occurs in all reruns, syz-verifier creates a report for the program.

I agree to abide by the anti-harassment policy

I agree

Primary author: MIHALI, Mara

Co-authors: ELVER, Marco (Google); VYUKOV, Dmitry (Google)

Presenters: MIHALI, Mara; ELVER, Marco (Google); VYUKOV, Dmitry (Google)

Session Classification: Testing and Fuzzing MC

Track Classification: Testing and Fuzzing MC