Ambiguity resolution of symbolic execution bug paths

Tibor Brunner, Péter Szécsi, Zoltán Porkoláb

Eötvös Loránd University
Department of Programming Languages and Compilers
bruntib@caesar.elte.hu, szepet95@gmail.com, gsd@caesar.elte.hu

Today's software systems tend to be more extensive and complex. It is inevitable that these systems contain some bugs which have to be fixed at the lowest level in the source code. Fortunately there are static analysis tools which aid the bug-finding process. Moreover there is a pursuit in programming language evolution so the languages become easily testable and analyzable.

Clang Static Analyzer [1] is a powerful tool which uses symbolic execution for finding erroneous code in software systems written in C, C++ or Objective-C. The output of the analyzer is a sequence of statements which define an execution path that leads to a software bug. The list of bug paths [2] is then presented to the developers who fix them or mark them false positive. Evaluating the reports requires a huge effort. In this article we classify the bug paths by their characteristics in order to be able recognizing some of these as multiple appearance of the same bug. This way we can reduce the number of presented bugs without loosing true positive reports.

References

[1] https://clang-analyzer.llvm.org/

[2] Hampapuram, Hari and Yang, Yue and Das, Manuvir *Symbolic Path Simulation in Path-sensitive Dataflow Analysis* ACM SIGSOFT Softw. Eng. Notes, 2006, p. 52–58.