LLFI: An LLVM-based Fault Injection Tool

LLFI is a fault-injection tool built using the LLVM compiler infrastructure. Fault injection is the process of systematically perturbing the program's execution according to a fault model, and observing the effects. The goal is to study the error resilience of the program under faults. LLVM is a compiler infrastructure for writing custom program analyses and transformations. It compiles programs in high-level languages such as C and C++ to a low-level IR. The IR preserves type information from the source-level, but at the same time, represents the detailed control and data flow of the program. We implement LLFI as a pass in LLVM that operates on the LLVM IR, and instruments it to inject specific kinds of faults.

Because fault-injection typically needs to be done thousands of times, it is desirable to avoid going through the compile cycle every time a new fault is to be injected. To this end, LLFI inserts special functions into the IR code of the application, and defers the actual injection to runtime. The functions provide both fault-injection and tracing of the execution after injection. This way, it is possible to inject as many faults as one wants without recompiling the application. Further, it is possible to defer the decision of the specific kind of fault to inject at runtime, depending on the runtime state of the application. Finally, it is possible to leverage the rich set of program analyses in LLVM in deciding where to inject the fault, thus allowing programmers to inject faults in selected portions of the program that satisfy certain characteristics e.g., inject faults in malloc calls within nested loops.

One of the main challenges in fault injection is to trace the propagation of the fault in the program, and to map it back to the source code. This is essential for understanding the weaknesses in the program's fault handling mechanisms and improving them. LLFI allows users to trace the propagation of the fault after its injection as the program executes, and also to identify specific kinds of targets or sinks in the program that must be compared with the fault-free run, to determine whether they have been corrupted by the fault. For example, one can easily determine if a fault has propagated to a system call executed by the program. However, this feature of LLFI requires that the program is deterministic.

To summarize, the following are the main features of LLFI:

1. Ability to inject faults into a wide range of applications written in many different languages (supported by LLVM), and to leverage the existing LLVM infrastructure for choosing the instructions to inject. We have currently tested LLFI on C and C++ applications only.

2. Ability to control at runtime, the kinds of faults that must be injected and where to inject them in the program. This determination can be made by examining the program's runtime state, and by considering source-level characteristics of the program's code.

3. Ability to trace the propagation of a fault, after injecting it, and examining the state of selected program elements in relation to their fault-free state. The tracing can be done at the level of LLVM IR, which is easier than doing it on the machine code.