

Explanation of How LLFI Works -- factorial

A. Preparation

1. Using example program: `/test_programs/factorial`.
You need to build linked .bc file: *factorial.bc*.
2. Input file: *input.factorial: 6*
3. Correct output: *720*

B. Injection

./run_all.sh factorial input.factorial ../../Debug+Asserts/lib 1000

After disassembling the *factorial.final_inject.bs.bc*, you can read the instrumented IR.

Below figure is part of the generated CFG of *factorial.final_inject.bs.bc*, the whole CFG is located in `/test_programs/factorial/cfg.main.pdf`:

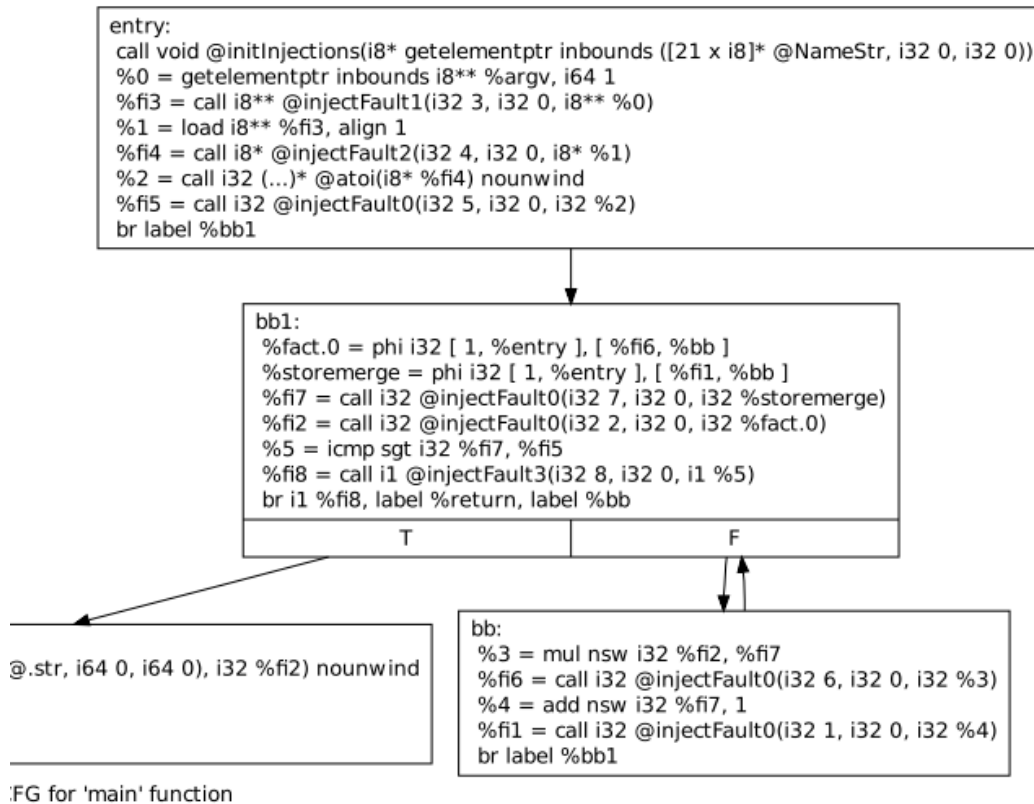


Figure 1 Parts of the control flow graph and IR after injection instrumentation

C. Output analysis

1. For one case: **id:count = 2:15**

size_byte=4 address=0x7fffdc305900 bytepos=2 bitpos=5 old errorbuf=0x0

Injected Fault : ID = 2 size = 32 old=0x0 new=0x20 count=15

llvm-dis factorial.final_inject.bs.bc -o factorial.final.ll

In factorial.final.ll, we can find %fi2 = call i32 @injectFault0(i32 2, i32 0, i32 %fact.0). When count is 15, the value of %fact.0 should be 2. Yet %fi2 replaces %fact.0 with value 2097154(0x00200002). In this way, the output becomes: $2097154 * 3 * 4 * 5 * 6 = 754975440$.

2. The overall statistics: *factorial_faultoutcome_stats.txt*

The statistic table:

Table 1 Output statistic table

APPLICATION	SDC	CRASH	BENIGN	HANG
factorial	817	50	129	4

This table indicates there are 817 SDCs, 50 crashes and 4 time-out results in 1000 times of fault-injection. The left 129 times result in benign output.