Assignment 2

Due date: Sunday, November 24

Objective: Find bugs in the SCIP Optimization Suite.

SCIP

- SCIP is a solver for optimization problems

  (in the context of this course, we don’t need to understand what it does)

- Open source project (https://scipopt.org)

- 1.4 M lines of code (~90% in C, plus some C++)

- first release: 2008

SCIP vs. GLPK

- GLPK is another open-source solver for optimization problems.
- SCIP is typically 2-10x faster
- SCIP is harder to install
  - “`apt-get install scip`” not available yet
  - “`brew install scip`” was just added recently
  - until 2023, SCIP was source-available
  - open-source since then (Apache license)

Problem

- `scip` reads a mathematical problem described by an input file
  (multiple formats accepted, we focus on the MPS and LP formats)

- then it solves the problems, prints messages, and writes solutions to an output file.

The format parsers contain bugs.

- Find them.

- Fix them.

LP format

```
Minimize
  cost: x + y
Subject To
  constraint_0:  3 x +   y >= 5
  constraint_1:    x + 2 y >= 3
Bounds
  x >= 0
  y >= 0
End
```

MPS format

```
NAME          example_problem
ROWS
 N  cost
 G  constraint_0
 G  constraint_1
COLUMNS
    x             cost                      1
    x             constraint_0              3
    x             constraint_1              1
    y             cost                      1
    y             constraint_0              1
    y             constraint_1              2
RHS
    rhs           constraint_0              5
    rhs           constraint_1              3
ENDATA
```

Approach

1. gather sample input files (MPS and LP)

2. compile SCIP with fuzzer instrumentation (using the AFL++ fuzzer)

3. fuzz SCIP

4. understand and fix the bugs

Rules

- Individual assignment:
  - Everyone must find problematic input files on their own computer

    you must compile SCIP, and run the fuzzer on your own device
  - Everyone must write their own explanation for bugs and fixes

- But:
  - You are allowed to help each other for compiling things and for running the fuzzer
  - You are encouraged to share hints on overcoming difficulties
  - You are strongly encouraged to ask me whenever you get stuck

Assertions

You are asked to find bugs that cause **crashes** (in “release” builds)

not just error messages.

The SCIP code contains many `assert()`s (enabled in “debug” builds only)

$\Rightarrow$ two possible angles:

- fuzz a release build, looking for crashes
- fuzz a debug build, looking for crashes and assertion failures
  - check that assertion failures would lead to a crash in a release build