COM S 413/513 Project 2: Comparing afl and klee **1**
Learning Objectives 1
Description 1
Deliverables (28 pt + extra credit) 2

COM S 413/513 Project 2: Comparing afl and klee

Learning Objectives

1. Strengthen the understandings of *fuzzing* and *symbolic execution*
2. Get hands-on experience with the-state-of-the-art tools
3. Improve problem solving skills on testing
4. Work with real-world software and bugs

Description

In this homework, we are going to run and further study the testing tools learned in our class. American fuzzy lop (afl) is a fuzzing tool that has found many bugs in real-world software. Klee is a symbolic execution tool that can automatically generate test inputs for covering as many branches as possible. We will use a few buggy programs to test and compare the performance of the two tools. In the following, please find a list of steps to follow:

(1) Install american fuzzy lop (afl): http://lcamtuf.coredump.cx/afl/
- Download afl.
- Read QuickStartGuide in the doc folder
- Test an example “test-instr.c”

(2) Install klee https://klee.github.io/
- Install “the docker version of klee” by following instructions: https://klee.github.io/docker/
- Go to klee_src/examples and run get_sign.c example following the tutorial in https://klee.github.io/tutorials/testing-function/ (small correction: under “Replaying test case” using clang instead of gcc) -- sudo access password is klee

(3) Compare afl and klee on the *get_sign* example
- Modify *get_sign* example to make it work for afl
- Introduce a bug to *get_sign.c* and test the buggy version of get_sign.c with afl and klee

(4) Compare afl and klee on *regexp.c* example provided by klee:
https://klee.github.io/tutorials/testing-regex/
- Modify *regexp.c* example and make it work for afl
- Introduce two buggy versions of regexp.c by implanting two bugs, test regexp.c with both afl and klee

(5) Compare afl and klee on a real-world program found on the open source repositories (tip: since klee is hard to set up, consider starting with small programs like gzip-1.2.4, ncompress or programs that work with klee, e.g., *programs from coreutils.*)
(6) Write-up your studies

Deliverables (28 pt + extra credit)

Please zip the following files and submit the zipped file to canvas under the "project 2: comparing afl and klee" column.

From Step 3, you'll submit (7 pt):

1. (2 pt) a modified version of get_sign.c for afl; screenshots to show that get_sign.c ran successfully with klee and afl
2. (1 pt) a buggy version of get_sign.c and a readme file that explains where is the bug and what is the bug
3. (2 pt) a folder that contains the test inputs generated from afl and klee
4. (2 pt) a folder that stores the output of running these test inputs on afl and klee

From Step 4, you'll submit (8 pt):

1. (2 pt) a modified version of regexp.c; screenshots to show that *regexp.c* ran successfully with klee and afl
2. (2 pt) two buggy versions of regexp.c and a readme file that explains where are the bugs and what are the bugs
3. (2 pt) a folder that contains the test inputs generated from afl and klee
4. (2 pt) a folder that stores the output of running these test inputs on afl and klee

From Step 5, you'll submit (8 pt + extra credit):

1. (1 pt) source code of the open source program
2. (3 pt) screenshots to show that the program works with klee and afl, modifications of klee and afl if any (you can explain the modifications and attach the relevant files)
3. (2 pt) a folder that contains the test inputs generated from afl and klee
4. (2 pt) a folder that stores the output of running these test inputs on afl and klee
5. (extra credit: 2pt per bug) explain any bugs you found, e.g., where they are located. Please include the description in reademe.txt

Submit a report that summarizes your studies (5 pt). You can use the following questions as a guidance.

(1) How many test inputs are generated by klee and afl respectively?
(2) How many crashes and hangs reported by afl and klee for the 3 programs you experimented with?
(3) Are these crashes and hangs related to the same bugs or different bugs?

(4) Given a fixed amount of time (e.g., 30 min or 1 hour), which tools find more crashes and bugs?
(5) Which tools find first crashes and bugs quickly?
(6) What are the advantages and disadvantages of afl and klee?

The homework is due Mar 10, Wed 11:59pm. Start early!