COM 402 exercises 2023, session 7: Fuzzing

Exercise 7.1

What is coverage-guided fuzzing, and how does it differ from black-box fuzzing?

Exercise 7.2

Given the following program, how likely is it for a blackbox, greybox, or whitebox fuzzer to find the correct value for `user_input`?

```
int foo(uint32_t user_input) {

        /* ... */

        if (user_input == CONSTANT) {
                /* ... */
                crash();
                /* ... */
        }

        /* ... */

        return 0;
}
```

Exercise 7.3

What is a sanitizer and why is it helpful in fuzzing? Why can certain bugs only be detected when using sanitizers?

Solutions to the Exercises

Solution 7.1

Coverage-guided fuzzing is a form of greybox fuzzing where the feedback metric is coverage profiles. The fuzzer implements a search algorithm to maximize the objective function which is code coverage. The fuzzer then leverages this feedback to select interesting inputs and mutate them, in an attempt to create even more interesting inputs. This differs from blackbox fuzzing in that the latter does not incorporate any feedback into the fuzzing process; inputs are sampled randomly with static probabilities that are not affected by the behavior of the program.

Solution 7.2

Assuming a blackbox fuzzer, the likelihood of randomly generating a 32-bit integer to match a specific value is 1 in 2^{32}. A more advanced fuzzer that leverages feedback could observe that such a comparison is made and then reinsert the constant into the input in the next iteration, increasing the likelihood to 1 in 2. A whitebox fuzzer knows of this condition beforehand (e.g. through symbolic execution) and can generate a valid input in a single attempt.

Solution 7.3

A sanitizer is a fault detector and reporter. It implements a security policy and enforces it by inserting runtime checks and raising a red flag when the policy is violated. They increase the likelihood that a bug is detected, which is a requirement for successful fuzzing. A fuzzer can only report/save a bug-triggering input if it is informed of the fault that happened, and sanitizers are a great way to do that. The most basic form of fault detection is through crashes: a bug corrupts memory and results in an invalid memory access, which the CPU detects and raises an exception for. However, not all memory corruptions crash, and not all bugs corrupt memory, so many of them remain undetected, unless a sanitizer which implements a policy that detects them is employed.