Generating test cases for a Blockchain REST API

Lisette Veldkamp

October 14, 2020

This document contains some background information regarding black-box testing and grammer-based fuzzing.

1 Research Questions Thesis

1. How effective is grammer-based fuzzing in terms of fault detection rate in a black-box Blockchain API setting?
2. What level of test coverage can be achieved using this technique?
3. How efficient (in terms of required execution time) is this technique?
4. Which vulnerabilities are detected with this technique?

2 Black-box test generation

The thesis research will focus on automatically generating test inputs for components without source code ('black-box').

Henard et al. [1] have compared white-box and black-box regression test prioritization approaches in terms of execution time and fault detection rate. They found that Combinatorial Interaction Testing (cover the maximum interactions between model inputs) and diversity-based techniques Input Model Diversity (maximize the NCD distance between inputs) and Input Test Set Diameter (maximize the distance between multisets of inputs) perform best among the black-box approaches.

3 Fuzzing

Fuzzing is an automated software testing technique that involves providing invalid, unexpected, or random data as inputs to a computer program. The program is then monitored for exceptions such as crashes, failing built-in code assertions, or potential memory leaks. Essentially, fuzzing is a dynamic testing technique that is based on the idea of feeding random data to a program "until it crashes." There are two main fuzzing techniques: mutation-based fuzzing (generate new variants based on existing data samples) and generation-based fuzzing (use a model of the input data or the vulnerabilities) [2].

An application of fuzzing in the context of test case generation of APIs is RESTler [3], a stateful REST API fuzzer. Each test is defined as a sequence of requests and responses, hereby exploring service states. RESTler generates tests by inferring dependencies among request types and by analyzing dynamic feedback from responses that were observed during prior test executions in order to generate new tests. RESTler is able to automatically generate sequences of requests (starting from the Swagger specification) that exploit the business logic exposed by the API in as tateful manner and without pre-recorded HTTP traffic.

3.1 Grammer-based fuzzing

Using a grammar-based specification of valid inputs is an example of generation-based fuzzing. This means having a grammar-based specification of valid inputs for an API.

References

[1] Christopher Henard, Mike Papadakis, Mark Harman, Yue Jia, and Yves Le Traon. Comparing white-box and black-box test prioritization. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, pages 523–534. IEEE, 2016.

[2] John Hoopes. *Virtualization for security: including sandboxing, disaster recovery, high availability, forensic analysis, and honeypotting.* Syngress, 2009.

[3] Vaggelis Atlidakis, Patrice Godefroid, and Marina Polishchuk. Restler: Stateful rest api fuzzing. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 748–758. IEEE, 2019.