Master Thesis Topic

Grammar-Based Fuzzing of Workflow Engines

Motivation and Background

Workflow engines (e.g., Nextflow [1]) are an integral part in scientific computing [2]. Workflow engines take task descriptions in a domain specific language, parse them, and then execute the corresponding workflow. As most software, the compilers for the domain specific language may contain bugs, which is why we want to test them. Grammar-based fuzzing [3-5] is a testing technique which uses grammars, to generate complex inputs. These inputs are then used to test the program. JQF [6] is a fuzzing framework which enables coverage-guided fuzzing through imperative *generators* that randomly sample syntactically valid inputs (e.g., inputs that conform to an input grammar).

Goals

The goal of this thesis is to use grammar-based fuzzing to test a workflow engine, specifically Nextflow, by implementing an input generator in JQF. The generator may be enhanced by custom, domain-specific input mutation operators, similar to existing grammar-based fuzzers like MoFuzz [5]. The thesis will test the engine and evaluate the efficacy and effectiveness of grammar-based fuzzing for finding bugs in workflow engines.

Description of the Task

The specific tasks are:

- Creating a model for the Nextflow DSL
- Familiarizing with JQF
- Evaluating the results against Nextflow

Research Type

Theoretical Aspects: **
Industrial Relevance: ****
Implementation ****

Prerequisite

The student should be enrolled in the bachelor of computer science program, and has completed the required course modules to start a bachelor thesis (or similar).

Skills required

Programming skills in Java, understanding of, or willingness to learn, the software engineering methods (like fuzz testing) and tools (e.g., JQF) needed for the project.

Contacts

Jan Arne Sparka (sparkaar@informatik.hu-berlin.de)
Software Engineering Group, Institut für Informatik, Humboldt-Universität zu Berlin

References

[1] Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., & Notredame, C. (2017). *Nextflow enables reproducible computational workflows.* Nature Biotechnology, 35(4), 316–319.
[2] Leser, U., Hilbrich, M., Draxl, C. *et al.* The Collaborative Research Center FONDA. *Datenbank Spektrum* **21,** 255–260 (2021).
[3] Godefroid, P. Fuzzing: Hack, art, and science. Communications of the ACM, 63(2), 70-76.
[4] Wang, J., Chen, B., Wei, L., & Liu, Y. Superion: Grammar-aware greybox fuzzing. *ICSE 2019*.
[5] Nguyen, H. L., Nassar, N., Kehrer, T., & Grunske, L. Mofuzz: A fuzzer suite for testing model-driven software engineering tools. ASE 2020.
[6] Rohan Padhye, Caroline Lemieux, and Koushik Sen. JQF: coverage-guided property-based testing in Java. *ISSTA 2019*.