Muse Design Document

Amit Seal Ami, Liz Weech, Yang Zhang

Motivation

Muse (or μSE) is a mutation-based soundness evaluation framework, which systematically evaluates Android static analysis tools to discover, document, and fix flaws by leveraging the well-founded practice of mutation analysis. Muse was originally written by Richard Bonnett in Fall 2017, and the motivation of this design document is to sketch out a refactoring of Muse so that it better reflects the design comments in its academic paper.

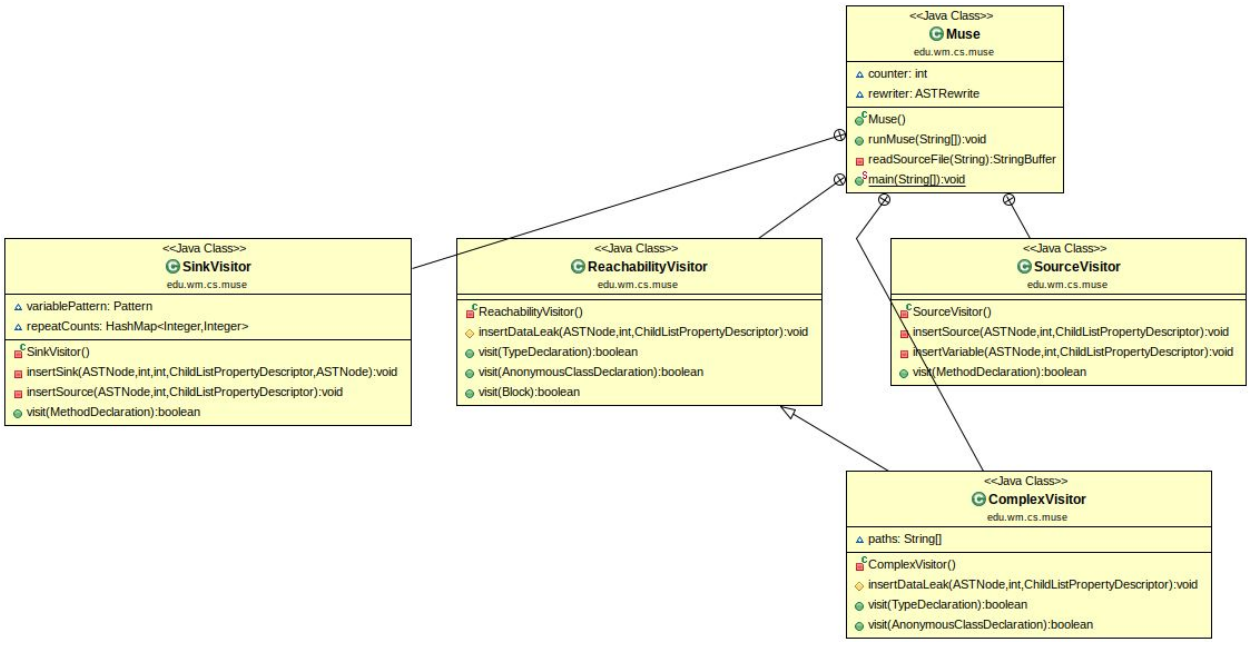
Key Terms

Security Operator: a description of the unwanted behavior that the security tool being analyzed aims to detect. For example, data leaks.

Mutant: code that represents the target unwanted behavior

Mutation Scheme: the specific methods for choosing where to apply security operators to inject mutations within Android apps.

Current Implementation Class Diagram



New Implementation Details

The MuSE application will now be further split into two classes for ease of interpretation and to follow the USENIX Security 18 project paper in greater detail. Those two new classes will be the Security Operator and Mutation Scheme. The following will only be a brief overview of the new classes, and further clarification should be found in the documentation of the code base.

Security Operator: The security operator class contain the basic properties from which all mutants derive from. Depending on the application, mutants may be altered to focus on specific goals the base application intends to cover.

Mutation Scheme: The new mutation scheme class encompasses all of the different types of mutation schemes MuSE will select from depending on the application to appropriately inject mutants in all possible and relevant locations in the app code.

[still under work]

Questions?

Contact us!
Amit: aami@email.wm.edu
Liz: eaweech@email.wm.edu
Yang: yzhang51@email.wm.edu