Prabhat Gehlot
Eduardo Preciado
Stephanie Contreras
Imon Daneshmand
Timothy Spengler

Project Report: JavaScript Spider Monkey

Unit Testing with Sufficient Coverage

- Unit Testing is not applicable to the project. An emphasis on automated testing was discussed.

Automated Testing

- Program Generator
    - https://github.com/csun-comp587-s20/jsfuzzing/blob/master/grammar.py
- Randomization
    - https://github.com/csun-comp587-s20/jsfuzzing/blob/8824d1c6be467a8bff563ad5c7bcbd57a96c5c45/grammar.py#L4
- Bound-Exhaustive
    - https://github.com/csun-comp587-s20/jsfuzzing/blob/8824d1c6be467a8bff563ad5c7bcbd57a96c5c45/grammar.py#L163

Lessons Learned

- Start with a smaller grammar because our initial grammar was more tedious to work with without actually improving the testing capabilities.
- Difficult to use randomness and still produce meaningful and interesting programs.
- I would choose a code repository that is easier to run tests on.
- Incorporating property testing could have been useful for the large code base.

Running the tests on spidermonkey.

- Once spidermonkey is compiled, we can use the jit-tests module to run the tests by copying the output from the entry_point to one of the tests folders. Following the spidermonkey test description, a result of 0 from JS shell implies that the shell ran the program without any errors, meaning no crash detected.