UTA

THE UNIVERSITY OF TEXAS
AT ARLINGTON

CSE 6324

Christoph Csallner

Title: Smart Contract Analysis tool - Slither and its defect fixing

Shubham Arun Malankar	--------1002031033
Nageshwar Ramkumar Jaiswal	--------1002033432
Ravi Prakasha	--------1002026832
Navyashree Budhihal Mutt	--------1001965572
Rushikesh Mahesh Bhagat	--------1001911486

Vision

The objective of the project is analysing the Slither smart contract static tool, inspect each functionalities, followed by discovering and mitigating new vulnerabilities which in turn makes the tool better efficient.

Feature and Workflow

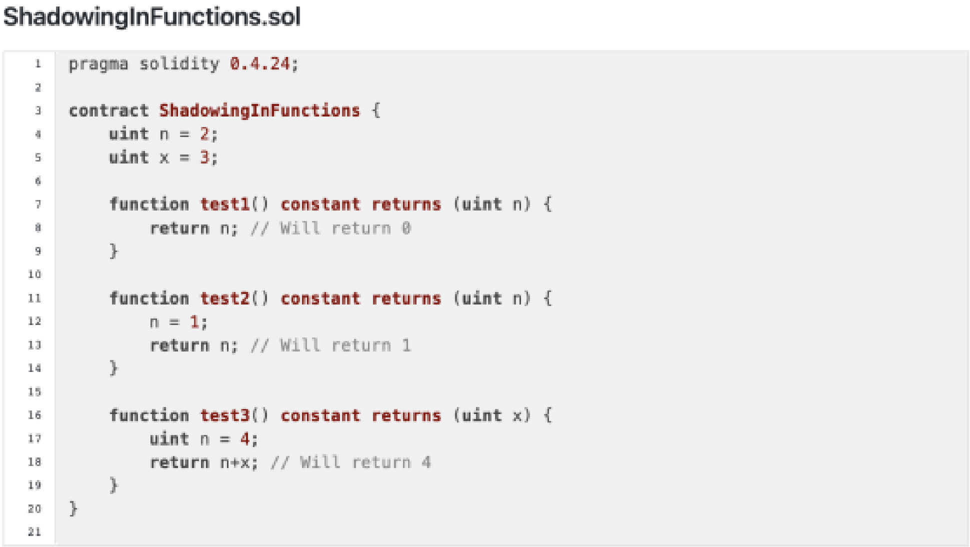
- Iteration 1 objective :
  - Analyze the static analysis tool - Slither, define our own understanding of each and every features built in it. Also, try to inspect solidity usage in the tool.
  - Mitigate SWC 119 - Shadowing the state variable, and deploy into the Slither tool.[5]
  - Solve the problems and errors caused post the previous step.

- Iteration 2 objective :
  - As the SWC - 119 is relatively to cause security issues as there is multiple definition on same variable, we may cause many inter-dependency errors, hence we will inspect and solve them. [5]

- Iteration 3 objective :
  - Implement any another vulnerability related to coding standards of the tool.
  - Test the final code efficiency.[2]

SWC-119 - Shadowing State Variables

- Solidity uses an unusual behaviour when using state variables during inheritance. [5]
- Contract B, which also has a declared state variable named x, might inherit contract A with a variable named x. [5]
- As a result, there would be two different versions of x, with access to one from contract A and the other from contract B. [5]



ShadowingInFunctions.sol

```
pragma solidity 0.4.24;

contract ShadowingInFunctions {
    uint n = 2;
    uint x = 3;

    function test1() constant returns (uint n) {
        return n; // Will return 0
    }

    function test2() constant returns (uint n) {
        n = 1;
        return n; // Will return 1
    }

    function test3() constant returns (uint x) {
        uint n = 4;
        return n+x; // Will return 4
    }
}

```

[5]

SWC 119 - Workflow

1) Reading the documentation thoroughly .

1) Understanding the code base.

1) Determining the file within the code base where we need to fix the vulnerability.

1) Work on building and developing the functional snippet.

1) Solving the araised errors and testing the final code on real time smart contracts.

Competitors

Category	Slither	Solium	MantiCore
Code Analysis	Static Analysis [3]	Static Analysis [3]	Static Analysis and Dynamic Analysis [3]
Input Provided	Solidity code as Input [2]	Solidity code file to perform linting and fix its issues [6]	Solidity code as Input [6]
Approaches Used To Find Vulnerability	Symbolic Execution , Control Graph and Data Flow Analysis [3]	Linting Rules, Automated Tools, code review [6]	Dynamic Symbol Execution [5]
Terminal Based User Interface	Yes [7]	No [7]	Yes [7]
Infrastructure	Python [2]	Javascript [7]	Python[6]

Risks

- Depending only on Slither to recognize potential security weaknesses in smart contract code can create a misleading belief in security since it may not uncover all issues. [3]
- Different version of smart contracts work differently on solidity compiler producing different issues, which makes it non-flexible for execution. [3]
- The analysis of the Slither is quite difficult as we have limited amount of precise and latest documentation. [3]

Customers

- Smart contracts are used by businessman and traders of blockchain and cryptocurrency. [1]
- Banks and large financial platforms are also the major customers of smart contracts.[1]
- For example, a easy vending machine is the best implementation of smart contract.

GitHub Repo Link

Link:https://github.com/shubhammalankar/ASE-CSE-6324-Team-5-Slither

Version: 0.1s

Acknowledgements

- Shovon Niverd

References

- [1] “Blockchain smart contracts: Applications, challenges, and future trends ”, Shafaq Naheed Khan, Faiza Loukil, Chirine Ghedira-Guegan, Elhadj Benkhelifa, Anoud Bani-Hani, 2021.
- [2] https://github.com/crytic/slither/wiki/SlithIR
- [3] https://arxiv.org/pdf/1908.09878.pdf
- [4] https://arxiv.org/pdf/1907.03890.pdf
- [5] https://swcregistry.io/
- [6] https://github.com/trailofbits/manticore
- [7] https://solium.readthedocs.io/en/latest/
- [8] https://github.com/duaraghav8/Ethlint

Thank you