CSE 6324 - 004 - Advance Topic in Software Engineering

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

Team 5

Team Members -

1. Shubham Arun Malankar(1002031033)
2. Nageshwar Ramkumar Jaiswal(1002033432)
3. Ravi Prakasha(1002026832)
4. Navyashree Budhihal Mutt(1001965572)
5. Rushikesh Mahesh Bhagat(1001911486)

Introduction:

Smart contracts are self-executing agreements with the terms of deal directly written into lines of code. These code-based contracts are stored on a blockchain platform and allow for the automation of high-value financial transactions between parties without the need for intermediaries[4]. Slither is a comprehensive Solidity static analysis framework designed to help developers identify vulnerabilities, improve code comprehension and perform custom analysis. This tool runs a set of vulnerability detection algorithms and provides a visual representation of the contract details. Additionally, it features an API that makes it simple to create custom analyses[6].

Project Vision:

The goal is to address the issues identified in the static analysis tool Slither by conducting a thorough analysis of smart contract source code and correcting any instances of non-compliance with established coding standards.

Static Code Analysis:

Static code analysis is used to identify potential bugs, security vulnerabilities, and other issues in the code without executing the code [3]. It can also check if the code adheres to industry standards, coding conventions, and best practices.[3]
There are different types of Static Analysis:[3]

1) Control analysis
2) Data analysis
3) Fault/failure analysis
4) Interface analysis

Slither

Slither is an open-source static analysis tool for Ethereum smart contracts. It is designed to help developers and security researchers identify potential security vulnerabilities and issues in their smart contracts. [5]

Features of Slither :

- Slither is a command-line tool that can be used to analyze Ethereum smart contracts, and its source code can be accessed on GitHub after implementation.
- Slither converts the entire contract code to SlithIR, its internal representation language.[1]
- Slither can detect code patterns that result in time-consuming code execution and deployment.[1]
- Slither generates a call graph that shows the relationships between functions, contracts, and libraries in a smart contract system.[1]
- Slither has over 80 detectors built-in to detect various types of security vulnerabilities and issues in Ethereum smart contracts.[1]

Competitors -

Solium -

1. Solium is a static analysis tool.
2. Input required is Solidity code file to perform linting and fix its issues.
3. Approach used to find vulnerability is Linting Rules, Automated Tools and code review.
4. Solium is not a terminal based interface
5. Solium tool is designed in Javascript

Manticore -

1. Manticore is a static as well as dynamic analysis tool.
2. Input required is Solidity code as Input
3. To find vulnerability approaches used in Manticore is Dynamic Symbol Execution.
4. It is a terminal based tool
5. Manticore is designed in python

Risk:

1. Reliance on the Tool: Relying solely on Slither to identify potential security vulnerabilities in smart contract code can lead to a false sense of security, as it may not detect all issues.
   Risk Exposure: The possibility of this risk occurring is estimated to be 40%. Additionally, it is estimated that if the risk does occur, it will have an impact of about 10 hours being spent on it. The risk exposure for this risk is calculated as the potential amount of extra time that may be needed to resolve the issue, which is estimated to be 4 extra hours.

2. Solidity version compatibility: Slither uses solidity compiler which requires solc files. Older versions of smart contracts do not get compiled in the updated Solidity compiler.
   Risk Exposure: This risk has a 50% probability of occurrence, with an estimated impact of 2 hours. This risk exposure is estimated to result in an additional 1 hour effort of resolving it.

3. No Additional Documentation for Slither Available: The documentation provided by Slither for installing the tools and understanding how the code is organized in each file is sparse and out-of-date. Furthermore, there is only one research paper available on the internet to understand the working and limitations of Slither.
   Risk Exposure: To reduce the above risk mentioned we can contact Trail of Bits for detailed information regarding installation and also about the tool . Furthermore, we can also find Youtube videos for the installation procedures for slither and refer to the research paper available on the Internet .

Feature:

The team's initial objective in their feature plan was to gain insight into the unique features of Slither. We attempted to comprehend the capabilities of the tools by analyzing their code structure but encountered difficulties in doing so as the code lacked adequate documentation in the form of comments, making it challenging to decipher. As a team, we are trying to compile and execute a sample smart contract with Slither; then exploit and fix vulnerabilities that are found.

Workflow:

Iteration 1 -

- Analyze the static analysis tool - Slither, define our own understanding of each and every feature 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.
- Solve the problems and errors caused post the previous step

Iteration 2 -

- As the SWC - 119 is relatively likely to cause security issues as there are multiple definitions on the same variable, we may cause many inter-dependency errors, hence we will inspect and solve them.

Iteration 3 -

- Implement any other vulnerability related to coding standards of the tool.
- Test the final code efficiency.

Customers:

1. Smart contracts are used by businessmen and traders of blockchain and cryptocurrency.
2. Banks and large financial platforms are also the major customers of smart contracts.

GitHub Repository:

Link:https://github.com/shubhammalankar/ASE-CSE-6324-Team-5-Slither
Version: 0.1

References -

[1] https://github.com/crytic/slither/wiki/SlithIR
[2] https://arxiv.org/pdf/1908.09878.pdf
[3]https://www.techtarget.com/whatis/definition/static-analysis-static-code-analysis
[4]https://blog.trailofbits.com/2019/05/27/slither-the-leading-static-analyzer-for-smart-contracts
[5] https://blog.trailofbits.com/2018/10/19/slither-a-solidity-static-analysis-framework/
[6]https://www.immunebytes.com/blog/slither-a-solidity-static-analyzer-for-smart-contracts/