CSE 6324 - 004

SolSecure Inception

Team 2

Members

Aishwarya Kalmangi - 1001988076
Darshan Shanthveer Patil - 1001719406
Diganth Prakash - 1001965135
FNU Nikitha Padmanabha - 1001968063
Vinay Kumar Shiva Kumar - 1001959006

Introduction

Simply put, smart contracts are blockchain-based algorithms that execute when certain criteria are met. They are often used to automate the implementation of an agreement so that all parties can be certain of the conclusion right away, without the need for an intermediary or additional delay. They can also automate a workflow such that when circumstances are met, the following action is executed. Simple phrases that are placed on a blockchain are how smart contracts operate. When predefined circumstances have been met, a network of computers will carry out the actions. [1]

GitHub Repository

- Link: https://github.com/darshanspatil07/SolSecure
- Version: v1.0.0

Project Vision

The major point of concern for the Smart Contract Technology is the handling of safety and reliability to avoid major financial losses.

Manticore
Manticore is a symbolic execution engine for binary analysis. It can be used to detect a variety of security-related issues in binary programs, including: Buffer overflows, Type confusion, Null pointer dereferences, etc.

Mythril
Mythril is a security analysis tool for the Ethereum Virtual Machine (EVM) bytecode. It can be used to detect various security vulnerabilities in smart contracts, including: Unchecked send, Uninitialized storage pointer, Call stack exhaustion, etc. [2][3]

The goal of combining Manticore and Mythril is to create a cutting-edge cybersecurity solution that takes advantage of both tools' strengths to provide organizations with a comprehensive defense against various types of attacks.
Manticore's sophisticated smart contract analysis framework can be combined with Manticore's advanced binary analysis capabilities to form a comprehensive and automated security assessment system. This system would allow organizations to easily, accurately, and quickly assess the security of their blockchain applications, smart contracts, and other critical assets.

This combination would also assist organizations in addressing the growing challenge of ensuring the security of their decentralized systems, which are increasingly being targeted by malicious actors.
A sophisticated solution that can find vulnerabilities, identify potential exploits, and offer real-time remedial recommendations will be created via the integration. A best-in-class cybersecurity solution that enables businesses to securely develop, implement, and manage secure decentralized systems is the ultimate goal of the merger of Manticore and Mythril.

Features

- **Integration with other tools** - The tool could offer the ability to analyze and assess the security of smart contracts written in Ethereum's Solidity programming language.

- **Taint Analysis** - The tool could include taint analysis capabilities, allowing it to track the flow of sensitive data throughout a smart contract and identify potential security vulnerabilities.

- **Scalability** - The tool could be designed to scale to accommodate large and complex smart contracts, ensuring that the analysis is both efficient and accurate.

- **Support for multiple programming languages** - The tool could offer support for multiple programming languages, beyond just Ethereum's Solidity language, allowing developers to analyze smart contracts written in other languages as well.

Software Development Plan

Combining Manticore and Mythril aims to produce a state-of-the-art cybersecurity solution that makes the most of each tool's advantages and gives enterprises a thorough defense against multiple forms of threats.
The timeline to complete the development is two months and the project will be complete in three iterations.

Iteration 1

This iteration will include the initial plan for development of SolSecure. As a part of it the user case/features to be implemented will be discussed in detail.

Create a resource plan that details the project's requirements, including the roles, duties, and expertise of the development team. The features to be implemented will be divided among team members.
We will begin with the initial setup to develop the first prototype. As a part of it Set up Linux WSL, download and install supporting libraries. The latest version of solidity will be installed to combine Mythril and Manticore.
Team will discuss the communication plan during development, then monitor progress of the project. When there are difficulties with implementation, assist one another.

Iteration 2

The group's goal for this iteration is to have the first working SolSecure prototype.
The team will decide which features are essential and which can be added after receiving a prototype, and they will work to implement those decisions.
Work to construct any new functionality that is necessary after selecting it. Keep a close check on the pending functionalities to be implemented and speed up development. [13]

Iteration 3

SolSecure testing will begin within this iteration. The group will choose a few of the already-in-use smart contracts to test, and each member will use SolSecure to do taint analysis, dynamic analysis, and symbolic execution in order to evaluate the effectiveness after all the necessary features have been included.Goal is to test all the test cases. After completion of SolSecure development, do comprehensive testing to find common smart contract flaws like reentrancy.In this final iteration the main goal is to have completed the prototype of SolSecure which combines Manticore and Mythril tools to identify vulnerabilities. To make sure SolSecure features and needs are fully met and the intended results are obtained, numerous tests will be conducted. [11]

Out of Scope:

We will try to implement a user interface for our tool if time allows, along with CLI functionality. Because UI is not our primary goal, we are listing it as an out of scope idea that can be worked on later.

Risks Management Plan

Top Possible Project Risks :

- **Compatibility problems**: It's possible that Manticore and Mythril weren't made to coexist in harmony, which could lead to problems that prohibit the tools from successfully integrating. [5]

- **Complexity of the integration process**: If the integration process is more difficult than anticipated, the implementation will take longer and cost more money.
- **Performance degradation**: Result from the integration of two complicated systems making the solution less effective and efficient.
- **Lack of Scalability**: As the number of assets and users increases, the combined system may find it difficult to scale, which could cause performance and stability problems.
- **Security Vulnerability**: New security flaws might be created during the integration process, and bad actors might take advantage of them to breach the system.
- **Technical debt**: The integrated system can contain technological debt that makes future maintenance and upgrades more challenging.

Mitigation Plan:

- **Testing** - Thorough testing is necessary to make sure the merged tool performs as intended and is free of any surprises before deploying it in production. This includes testing the functionality of the merged product as well as the integration of the two tools. [7]
- **Secure coding practices** - Utilize secure coding techniques when creating the combined tool to reduce the possibility of security flaws. When combining the two tools and writing the code for the final product, secure coding techniques must be used.
- **Regular Security Updates** - Update the integrated tool frequently to make sure it is shielded from the most recent security flaws and threats. This entails updating the code for the integrated tool as well as Manticore and Mythril.
- **Monitoring and response**: Keep an eye out for security vulnerabilities with the combined tool and have a plan in place for handling them when they do. This involves keeping an eye out for security flaws, potential exploits, and any other potential security problems.
- **Third-party security assessments**: Regularly perform third-party security assessments of the combined tool to identify and remediate security vulnerabilities. This includes performing security assessments of the code for the combined tool, as well as of the code for Manticore and Mythril. [8]

Risk Exposure:

- **Overly Optimistic Schedule**
  - Probability of Risk Occurrence = 50%
  - Effect of Risk Occurring = 15 hours (extra)

    - Risk Exposure = 7.5 hours
- **Performance Degradation**
    - Probability of Risk Occurrence = 60%
    - Effect of Risk Occurring = 18 hours (extra)
    - Risk Exposure = 10.8 hours

Constraints

- Scheduled delivery date - At the conclusion of each iteration, the written deliverables, presentations, and repo checks must be prepared for submission. Dr. Christopher Csallner, GTA Mohammed Rifat, and for review, Team 3 must be able to review the contribution right away. They will also grade its quality and provide recommendations on how to make it better for the following revision. Delays in submission would result in incomplete projects by the project's ultimate scheduled delivery date after delays in reviewing, grading, and feedback. In order to achieve better results, we also intend to conduct further internal reviews.

Competitors

Tool Name	Method	Description
SolSecure	**Dynamic analysis + Taint analysis + Symbolic execution**	**Solsecure is a blend of Dynamic Analysis, Taint Analysis, and Symbolic Execution approaches. SolSecure employs Manticore's Dynamic Analyses to uncover any gas related vulnerabilities, as well as Mythril as an API for writing custom analysis. Furthermore, SolSecure will employ MantiCore's Symbolic Execution for Input Generation and Error Discovery to create an overall analysis tool that will assist in identifying the vast majority of flaws and vulnerabilities.**
GasGuage	Static + Dynamic Analysis	GasGuage is a combination of Static and Dynamic Analysis that focuses on gas related vulnerabilities that ensures that Solidity gas does not surpass the gas limit during transactions. [6]
MantiCore	Symbolic Execution	MantiCore is a symbolic Execution tool for the analysis of Smart contracts. The main features of MantiCore include Program Exploration, Input Generation and Error Discovery.[9]

Slither	Static Analysis	Slither is a Solidity based Static Analyzer tool that detects vulnerabilities and provides an API to write custom analysis. [10]
Mythril	Symbolic Execution	Mythril utilizes Symbolic Execution and SMT solving to identify security vulnerabilities in smart contracts built for Ethereum.[12]

References

[1]https://www.ibm.com/topics/smart-contracts#:~:text=Smart%20contracts%20are%20simply%20programs,intermediary's%20involvement%20or%20time%20loss

[2]https://www.alchemy.com/dapps/manticore#:~:text=Released%20in%202017%20by%20TrailOfBits,states%20a%20program%20can%20reach

[3]https://medium.com/haloblock/how-to-install-mythril-a-smart-contract-security-tool-tutorial-4876991a823c

[4]Risk Assessment Analysis: Exposure and Target Accounts for Risk Hedging - FinanceTrainingCourse.com, accessed 09/11/2022

[5]Bistarelli, S., Mazzante, G., Micheletti, M., Mostarda, L., Sestili, D., & Tiezzi, F. (2020). Ethereum smart contracts: Analysis and statistics of their source code and opcodes. *Internet of Things*,

[6]SAICSIT '20: Conference of the South African Institute of Computer Scientists and Information Technologists 2020September 2020 Pages 35–43https://doi.org/10.1145/3410886.3410907

[7](2020, March 19). *Crypto Market Pool - Gas in Solidity smart contracts*. Crypto Market Pool. https://cryptomarketpool.com/gas-in-solidity-smart-contracts/, accessed 09/11/2022

[8]W. Zhang, S. Banescu, L. Pasos, S. Stewart and V. Ganesh, "MPro: Combining Static and Symbolic Analysis for Scalable Testing of Smart Contract," in 2019

[9]Mark Mossberg, Felipe Manzano, Eric Hennenfent, Alex Groce, Gustavo Grieco, Josselin Feist, Trent Brunson, and Artem Dinaburg. 2019. Manticore: a user-friendly symbolic execution framework for binaries and smart contracts

[10]Liu, Y., Xu, J., Cui, B. (2022). Smart Contract Vulnerability Detection Based on Symbolic Execution Technology. In: Lu, W., Zhang, Y., Wen, W., Yan, H., Li, C. (eds) Cyber Security

[11]https://www.cprime.com/wp-content/uplads/2020/09/Untitled-3.png, Accessed 09/11/2022

[12]https://lightrains.com/blogs/solidity-static-analysis-tools/ , Accessed 09/10/2022.

[13]https://ieeexplore.ieee.org/document/9762279, Accessed 09/11/2022