CSE 6324:Advanced Topics in Software Engineering

Vulnerability analysis using Conkas tool

Team 2:

Hema Sri Sesham(1001955263)

Moulin Chandrakant Jariwala(1001858654)

Parshva Urmish Shah(1001838879)

Vineeth Kumar Ananthula(1001953922)

TABLE OF CONTENT

Sr No.	Title	Page No
1.	Abstract	3
2.	Architecture	3
3.	Planned Features	4
4.	Comparison	4
5.	Target Users/Customers	5
6.	References	6

ABSTRACT

Modern blockchains like Ethereum, enable any user to create a program, called Smart Contract, and run-in distributed network. Ethereum also has a cryptocurrency called Ether and those Smart Contracts can send and receive Ether between each other via transactions. Once a Smart Contract is deployed, the owner of that contract can no longer make updates to fix it if someone finds a vulnerability. To try to reduce the probability of new future attacks happening Conkas was introduced, a modular static analysis tool that uses symbolic execution to find traces that lead to vulnerabilities and uses an intermediate representation. The users can interact with Conkas via Command-Line Interface (CLI) and the output will be the result of the analysis. Conkas supports Ethereum bytecode or contracts written in Solidity and is compatible with all versions of Solidity, but the analysis is done at the bytecode level. [1]

Conkas supports 5 modules that detect vulnerabilities: Arithmetic, Front-Running, Reentrancy, Time Manipulation and Unchecked Low-Level Calls. Conkas is easy to extend, meaning that you can add your custom modules to detect other types of vulnerabilities. We plan to improve conkas by adding Tx.Origin and costly loops feature in conkas.

ARCHITECTURE

Conkas uses a modular architecture. The users can provide byte-code and this byte code is passed to the rattle module, in this the symbolic execution engine is responsible to iterate and generate traces. These are provided to the Detectors module, in this one of the sub module is responsible to detect a category of vulnerability.

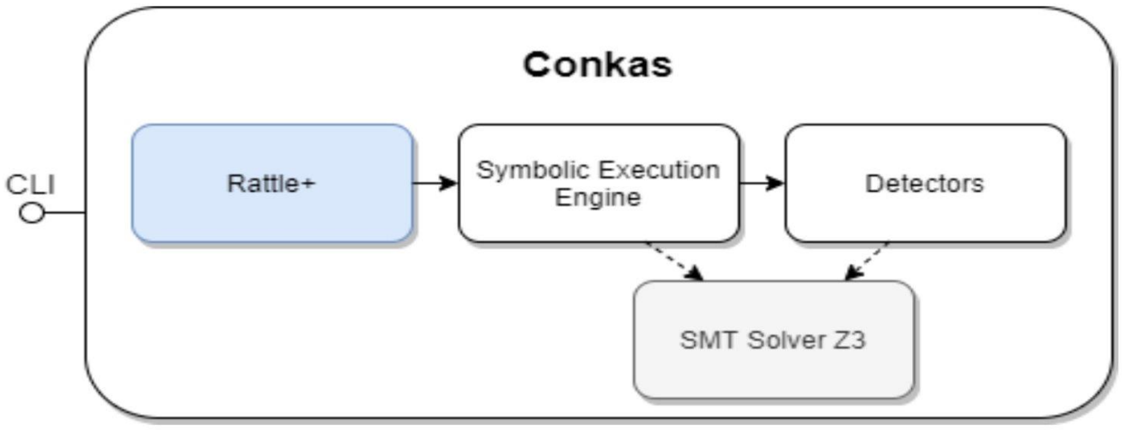

Figure 1: Architecture of Conkas tool.[1]

PLANNED FEATURES

We will be implementing the following two features phase wise to improve the conkas tool.

1. Tx.Origin

Risk involved with Tx.Origin : Tx.Origin is a global variable in Solidity which returns the address of the account that sent the transaction. Contracts that use the tx.origin to authorize users are vulnerable to phishing attacks. It is critical to never use tx.origin for authorization since another contract can contact a contract using a fallback function and get authorization because the permitted address is recorded in tx.origin. [2]. In June,2021 THORChain Org suffered this attack and lost around 8 million dollars. [3]

Dealing with Tx.Origin : Detecting the use of tx.origin variable will eliminate the vulnerability of getting phished.

2. Costly Loops

Risk involved with Costly Loops: An EVM is constrained by the block gas limit, and the cost of gas connected with contracts is one of the factors that encourage users to interact with your contracts. In Solidity, loops may be used for any function. However, if the loop involves updating certain contract state variables, it should be constrained; otherwise, the contract may get stuck if the loop iteration exceeds the block's gas limit. If a loop consumes more gas than the block's gas limit, the transaction will not be added to the blockchain and will revert. As a result, the transaction fails. As a result, it is important to avoid employing loops that alter contract state variables in the first place.

Costly loops and gas limit refers to an attacker including infinite loops within an array to exhaust the gas limit mimicking DoS attack and freeing the transaction.

Dealing with Costly Loops: We will be implementing a methodology to detect costly loops.

COMPARISON TO OTHER SMART CONTRACT VULNERABILITY DETECTION TOOLS

Several tools for smart contracts have been developed to address vulnerabilities detection. Other prominent tools that compete with this Conkas include Slither, Oyente, Osiris, Manticore, Mythril, and Securify, Smartcheck. Using the SmartBugs framework, Conkas' performance was examined compared to that of other tools. When the true positive rate of all tools was calculated, Conkas had the highest percentage.

Conkas false positives are decreasing in practically every area, but the arithmetic category is the most penalised. We have less false positives than Mythril, and we have the fewest false positives in Unchecked Low Calls.

Category	Conkas	Honeybadger	Maian	Manticore	Mythril	Osiris	Oyente	Securify	Slither	Smartcheck	Total
Access Control	0/24 0%	0/24 0%	0/24 0%	5/24 21%	4/24 17%	0/24 0%	0/24 0%	1/24 4%	6/24 25%	2/24 8%	8/24 33%
Arithmetic	19/23 83%	0/23 0%	0/23 0%	13/23 57%	16/23 70%	13/23 57%	18/23 78%	0/23 0%	0/23 0%	1/23 4%	22/23 96%
Denial Service	0/14 0%	0/14 0%	0/14 0%	0/14 0%	0/14 0%	0/14 0%	0/14 0%	0/14 0%	0/14 0%	1/14 7%	1/14 7%
Front Running	2/7 29%	0/7 0%	0/7 0%	0/7 0%	2/7 29%	0/7 0%	2/7 29%	2/7 29%	0/7 0%	0/7 0%	2/ 7 29%
Reentrancy	30/34 88%	19/34 56%	0/34 0%	15/34 44%	25/34 74%	21/34 62%	28/34 82%	14/34 41%	33/34 97%	30/34 88%	33/34 97%
Time Manipulation	7/7 100%	0/7 0%	0/7 0%	4/7 57%	0/7 0%	2/7 29%	0/7 0%	0/7 0%	3/7 43%	2/7 29%	7/ 7 100%
Unchecked Low Calls	62/75 83%	0/75 0%	0/75 0%	9/75 12%	60/75 80%	0/75 0%	0/75 0%	50/75 67%	51/75 68%	61/75 81%	70/75 93%
Other	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/5 0%	0/ 5 0%
Total	120/224 54%	19/224 8%	0/224 0%	46/224 21%	107/224 48%	36/224 16%	48/224 21%	67/224 30%	93/224 42%	97/224 43%	143/224 64%

Table 1: Results of Conkas compared to other tools

TARGET USERS/CUSTOMERS OF THE TOOL

The main users of Conkas are the web3 developers and smart contract auditors. Auditing a Smart Contract is about extensively looking through the code to avoid any data breaches or bugs. This is required because once this code is launched, it cannot be modified, which might result in massive security concerns, data loss, monetary loss and so on. Hence, it is essential to use an analysis tool to detect and eliminate all the vulnerabilities in a contract before it is deployed.

Project Github Repo: https://github.com/shahparshva72/CSE6324-Team2-Project

References:

[1] Veloso, Nuno. Conkas: A Modular and Static Analysis Tool for Ethereum Bytecode - ULisboa. Jan.2021,
https://fenix.tecnico.ulisboa.pt/downloadFile/1689244997262417/94080-Nuno-Veloso_resumo.pdf.

[2] Polak, Kamil. "Hacking Solidity: Contracts Using Tx.origin for Authorization Are Vulnerable to Phishing." *HackerNoon*, 16 Jan.2022,
https://hackernoon.com/hacking-solidity-contracts-using-txorigin-for-authorization-are-vulnerable-to-phishing

[3] Sebastian Sinclair. "Blockchain Protocol Thorchain Suffers $8M Hack" Coindesk, 14 Sep. 2021,
https://www.coindesk.com/markets/2021/07/23/blockchain-protocol-thorchain-suffers-8m-hack/