Ethereum Smart Contracts

Devansh Patel
Information System Security
Concordia University
Montreal, Canada
pa_devan@live.concordia.ca

Darshit Gajjar
Information System Security
Concordia University
Montreal, Canada
d_gajj@live.concordia.ca

Rushi Chandalia
Information System Security
Concordia University
Montreal, Canada
ru_chan@live.concordia.ca

Kamal Sharma
Information System Security
Concordia University
Montreal, Canada
kamal.sharma@mail.concordia.ca

Palak Kaur Sodhi
Information System Security
Concordia University
Montreal, Canada
p_sod@live.concordia.ca

Sachin Verma
Information System Security
Concordia University
Montreal, Canada
v_sachi@live.concordia.ca

Gulshan Joshi
Information System Security
Concordia University
Montreal, Canada
gu_joshi@live.concordia.ca

Varsha Vivek
Information System Security
Concordia University
Montreal, Canada
va_vivek@live.concordia.ca

Abstract— To provide flash loan to users, the Unstoppable and Naive Receiver contract was created. In this paper, we thoroughly examined both contracts and found a number of weaknesses that attackers could take advantage of. We discovered that the contract is particularly susceptible to denial-of-service and reentrancy attacks. We also suggested a number of mitigation techniques, such as building fail-safe methods and providing input validation, to address these issues. Overall, our findings emphasize the significance of carrying out exhaustive security evaluations and putting in place suitable security controls to guarantee the security of DeFi applications.

Keywords—smart contracts, ethereum, blockchain, vulnerability, code analysis, exploitation.

I. INTRODUCTION

A smart contract is a self-executing commitment with integrated lines of code that define the parameters of the agreement between both the contract's counterparties. A smart agreement serves as a digital replica of the traditional paper contract that actively maintains and carries out its conditions. The smart contract is performed over a blockchain system, and the network's various computers, each contain a copy of the contract's script. This guarantees a safer and more open implementation and execution of the contract terms. Furthermore, since the code of a smart contract is checked by every member of the blockchain network, smart contracts do not need an intermediary to be validated. The cost to counterparties is diminished significantly by eliminating the intermediary from the deal. Blockchain technology serves as the foundation for the notion of smart contracts.

A blockchain is a distributed network made up of a continuously expanding list of records (blocks) connected by encryption. Unlike a traditional database, a blockchain technology does not have a single central location. Each machine on the network has permission to view the information that is recorded in the blockchain. The network is consequently less susceptible to mistakes or attacks. A data on one device cannot be changed in a blockchain without also updating the identical information on other computers in the network. With a blockchain, transactions are organised into blocks that are connected by a chain. Only once the preceding block is finished is a new block formed. Each block comprises a cryptographic hash of the preceding block and is presented in a linear chronological sequence.

There are multiple steps involved in working of smart contracts. The contract's parameters should initially be decided by the counterparties. The completed contractual requirements are then converted into a computer program. In essence, the program contains a variety of conditional statements that outline several circumstances for a potential future transaction. As a piece of code is written, it is copied across the blockchain's users and saved in the network. The code is then compiled and executed by all machines on the network. The appropriate transaction is carried out if a condition of the contract is met and has been confirmed by every user of the blockchain network.

Code analysis plays a major role in this process. Basically, code analysis investigates programming code to discover potential pitfalls or mistakes, assure compliance with coding standards, and guarantee quality. Code analysis can be done statically (viewing the program without running it) or dynamically (executing it and evaluating its performance). There are several tools and methodologies for doing code review.