soken

SMART CONTRACT SECURITY AUDIT

Foundation

Scan and check this report was posted at Soken Github

July, 2023

Website: soken.io

Table of Contents

Disclaimer

This is a comprehensive report based on our automated and manual examination of cybersecurity vulnerabilities and framework flaws of the project's smart contract.

Reading the full analysis report is essential to build your understanding of project's security level. It is crucial to take note, though we have done our best to perform this analysis and report, that you should not rely on the our research and cannot claim what it states or how we created it.

Before making any judgments, you have to conduct your own independent research.

We will discuss this in more depth in the following disclaimer - please read it fully.

DISCLAIMER: You agree to the terms of this disclaimer by reading this report or any portion thereof. Please stop reading this report and remove and delete any copies of this report that you download and/or print if you do not agree to these conditions. Scan and verify report's presence in the GitHub repository by a qr-code at the title page. This report is for non-reliability information only and does not represent investment advice. No one shall be entitled to depend on the report or its contents, and Soken and its affiliates shall not be held responsible to you or anyone else, nor shall Soken provide any guarantee or representation to any person with regard to the accuracy or integrity of the report.

Without any terms, warranties or other conditions other than as set forth in that exclusion and Soken excludes hereby all representations, warrants, conditions and other terms (including, without limitation, guarantees implied by the law of satisfactory quality, fitness for purposes and the use of reasonable care and skills).

The report is provided as "as is" and does not contain any terms and conditions. Except as legally banned, Soken disclaims all responsibility and responsibilities and no claim against Soken is made to any amount or type of loss or damages (without limitation, direct, indirect, special, punitive, consequential or pure economic loses or losses) that may be caused by you or any other person, or any damages or damages, including without limitations (whether innocent or negligent).

Security analysis is based only on the smart contracts. No applications or operations were reviewed for security. No product code has been reviewed.

Procedure

Our analysis contains following steps:

1. Project Analysis;

2. Manual analysis of smart contracts:
   - Deploying smart contracts on any of the network(Ropsten/Rinkeby) using Remix IDE
   - Hashes of all transaction will be recorded
   - Behaviour of functions and gas consumption is noted, as well.

3. Unit Testing:
   - Smart contract functions will be unit tested on multiple parameters and under multiple conditions to ensure that all paths of functions are functioning as intended.
   - In this phase intended behaviour of smart contract is verified.
   - In this phase, we would also ensure that smart contract functions are not consuming unnecessary gas.
   - Gas limits of functions will be verified in this stage.

4. Automated Testing:
   - Mythril
   - Oyente
   - Manticore
   - Solgraph

Terminology

We categorize the finding into 4 categories based on their vulnerability:

- Low-severity issue — less important, must be analyzed
- Medium-severity issue — important, needs to be analyzed and fixed
- High-severity issue —important, might cause vulnerabilities, must be analyzed and fixed
- Critical-severity issue —serious bug causes, must be analyzed and fixed.

Limitations

The security audit of Smart Contract cannot cover all vulnerabilities. Even if no vulnerabilities are detected in the audit, there is no guarantee that future smart contracts are safe. Smart contracts are in most cases safeguarded against specific sorts of attacks. In order to find as many flaws as possible, we carried out a comprehensive smart contract audit. Audit is a document that is not legally binding and guarantees nothing.

Basic Security Recommendation

Unlike hardware and paper wallets, hot wallets are connected to the internet and store private keys online, which exposes them to greater risk. If a company or an individual holds significant amounts of cryptocurrency in a hot wallet, they should consider using MultiSig addresses. Wallet security is enhanced when private keys are stored in different locations and are not controlled by a single entity.

Token Contract Details for 04.07.2023

Contract Name: **FoundationToken**

Deployed address: **0x9c771503a42f9b6e498d89c43e99F2e58CB8B384**

Total Supply: **17,760,000,000,000**

Token Tracker: **SLD**

Decimals: **18**

Token holders: **5**

Transactions count: **6**

Top 100 holders dominance: **100.00%**

Audit Details

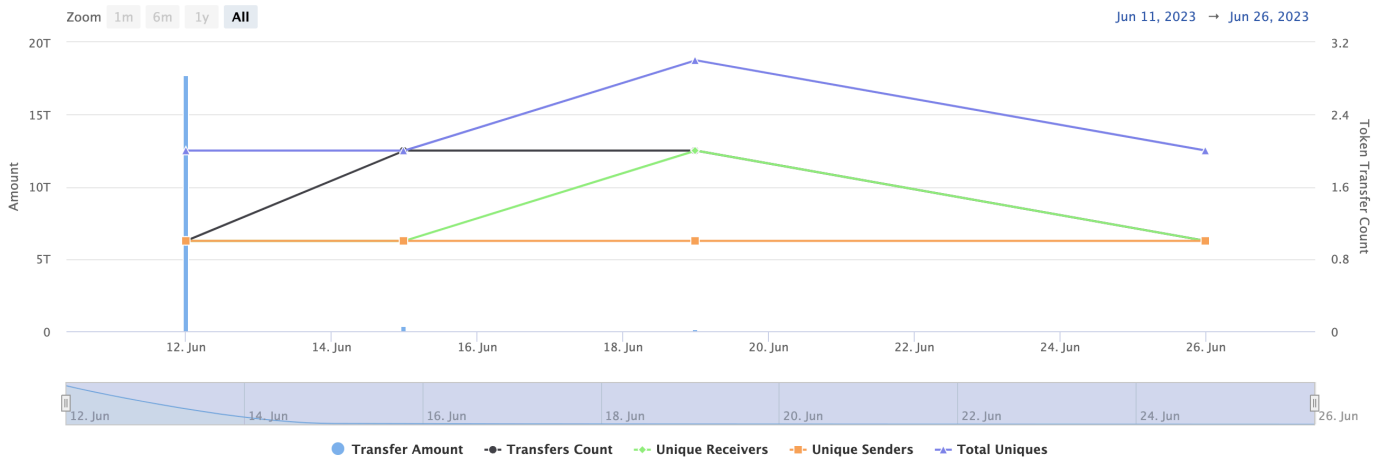
Project Name: **Foundation**

Language: **Solidity**

Compiler Version: **v0.8.15**

Blockchain: **BSC**

Token Analytics



Contract Function Details

- [Int] _msgSender
- [Int] _msgData
- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] transfer
- [Ext] allowance
- [Ext] approve
- [Ext] transferFrom
- [Ext] name
- [Ext] symbol
- [Ext] decimals
- [Pub] name
- [Pub] symbol
- [Pub] decimals
- [Pub] totalSupply
- [Pub] balanceOf
- [Pub] transfer
- [Pub] allowance
- [Pub] approve
- [Pub] transferFrom
- [Pub] increaseAllowance
- [Pub] decreaseAllowance
- [Int] _transfer
- [Int] _createInitialSupply
- [Int] _burn
- [Int] _approve
- [Pub] owner
- [Ext] renounceOwnership
- [Pub] transferOwnership
- [Ext] factory
- [Ext] WETH
- [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens
- [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens
- [Ext] addLiquidityETH
- [Ext] createPair
- [Ext] enableTrading
- [Ext] updateSwapTokensAtAmount
- [Ext] setAutomatedMarketMakerPair
- [Prv] _setAutomatedMarketMakerPair
- [Pub] excludeFromFees
- [Int] _transfer
- [Prv] swapTokensForEth
- [Prv] addLiquidity

- [Prv] swapBack
- [Ext] withdrawStuckETH
- [Ext] setOperationsAddress
- [Ext] forceSwapBack

Vulnerabilities checking

Issue Description	Checking Status
Compiler Errors	Completed
Delays in Data Delivery	Completed
Re-entrancy	Completed
Transaction-Ordering Dependence	Completed
Timestamp Dependence	Completed
Shadowing State Variables	Completed
DoS with Failed Call	Completed
DoS with Block Gas Limit	Completed
Outdated Complier Version	Completed
Assert Violation	Completed
Use of Deprecated Solidity Functions	Completed
Integer Overflow and Underflow	Completed
Function Default Visibility	Completed
Malicious Event Log	Completed
Math Accuracy	Completed
Design Logic	Completed
Fallback Function Security	Completed
Cross-function Race Conditions	Completed
Safe Zeppelin Module	Completed

Security Issues

1) Controlled low-level call: High-severity.
Status: Resolved.
Line: #563

The contract was using delegatecall() or call() which was accepting address controlled by a user. This can have devastating effects on the contract as a delegate call allows the contract to execute code belonging to other contracts but using it's own storage. This can very easily lead to a loss of funds and compromise of the contract.

Recommendation:

Do not allow user-controlled data inside the delegatecall() and the call() function.

2) Unchecked Transfer: High-severity.

Status: Acknowledged

Line: #478, #484

Some tokens do not revert the transaction when the transfer or transferFrom fails and returns False. Hence we must check the return value after calling the transfer or transferFrom function.

Recommendation:

Use OpenZeppelin SafeERC20's **safetransfer** and **safetransferFrom** functions.

3) Return Value of low-level calls: Medium-severity.

Lines: #557, #563

Status: Acknowledged

The functions do not check the return value of low-level calls. This can lock Ether in the contract if the call fails or may compromise the contract if the ownership is being changed. The following calls were detected without return value validations - call

Recommendation:

Ensure return value is checked using conditional statements for low-level calls. We should also ensure that we log failed calls using events.

Conclusion for project owner

High and medium-severity issues exist within smart contracts.

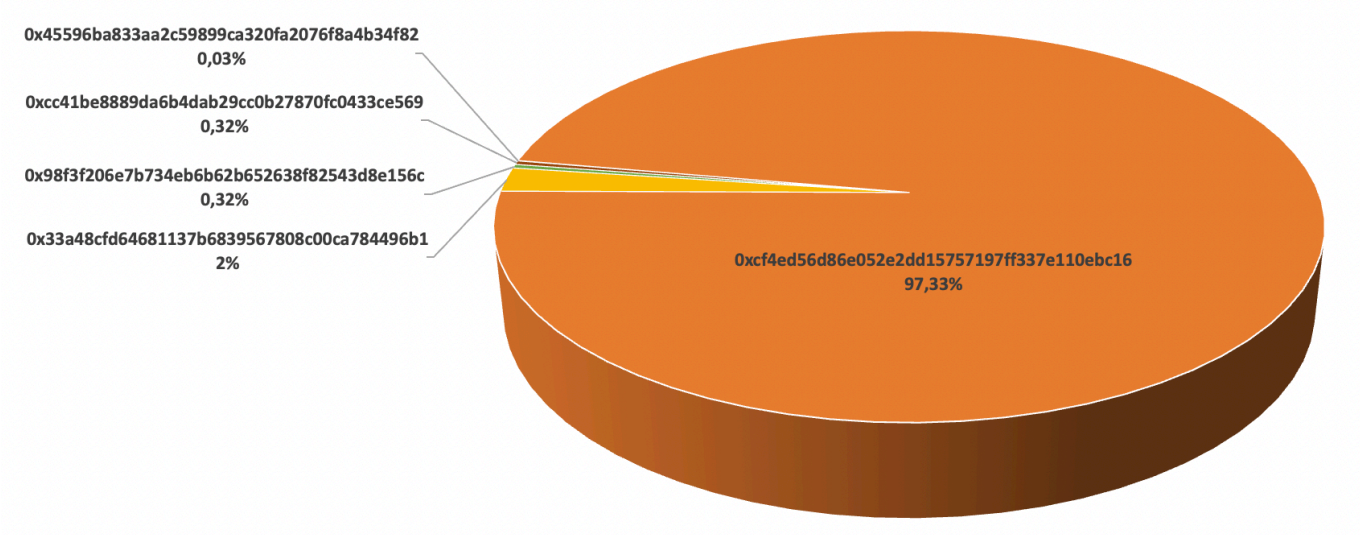
NOTE: Please check the disclaimer above and note, that audit makes no statements or warranties on business model, investment attractiveness or code sustainability. Contract security report for community

SECURITY REPORT
FOR COMMUNITY

Foundation

soken

SLD Token Distribution



SLD Top 10 Holders

Rank	Address	Quantity (Token)	Percentage
1	0xcf4ed56d86e052e2dd15757197ff337e110ebc16	17,286,318,888,891	97.3329%
2	0x33a48cfd64681137b6839567808c00ca784496b1	355,200,000,000	2.0000%
3	0x98f3f206e7b734eb6b62b652638f82543d8e156c	56,399,999,999	0.3176%
4	0xcc41be8889da6b4dab29cc0b27870fc0433ce569	56,389,999,999	0.3175%
5	0x45596ba833aa2c59899ca320fa2076f8a4b34f82	5,691,111,111	0.0320%

Soken Contact Info

Website: www.soken.io

Telegram: @team_soken

GitHub: sokenteam

Twitter: @soken_team