SAFUAUDIT

SMART CONTRACT SECURITY ASSESSMENT

PROJECT:

SIMBA

DATE:

JUNE 07, 2023

https://t.me/SafuAudit

www.safuaudit.com

Introduction

Client	Simba
Language	Solidity
Contract Address	0x11Ae0efA2FF44Bc6Eb33D41030eb01052fc5d6b7
Owner	0x31f77fcd1DBDe5c81F9A9631056600924246df7A
Deployer	0x31f77fcd1DBDe5c81F9A9631056600924246df7A
SHA-256 Hash	8436337b48f649a6e04581f902022f1577cbe6ac
Decimals	18
Supply	100000000000
Platform	Binance Smart Chain
Compiler	0.8.18+commit.87f61d96
Optimization	No with 200 runs
Website	https://simba.rocks/
Twitter	https://twitter.com/token_simba
Telegram	https://t.me/+FUD-RkA_xwhjNzU0

Overview

Fees

- Buy fees: 3%
- Sell fees: 3%

Fees privileges

- Owner can set fees up to 10%

Ownership

- Owned

Minting

- No

Max Tx Amount

- Can set maxTx for buying to any value, including 0

Pause

- Can't pause

Blacklist

- Can't blacklist

Other Privileges

- Owner can exclude/include from fees
- Owner can exclude/include from Max Tx

Table Of Contents

Risk Classification

Critical

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium

Issues on this level could potentially bring problems and should eventually be fixed.

Minor

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

Informational

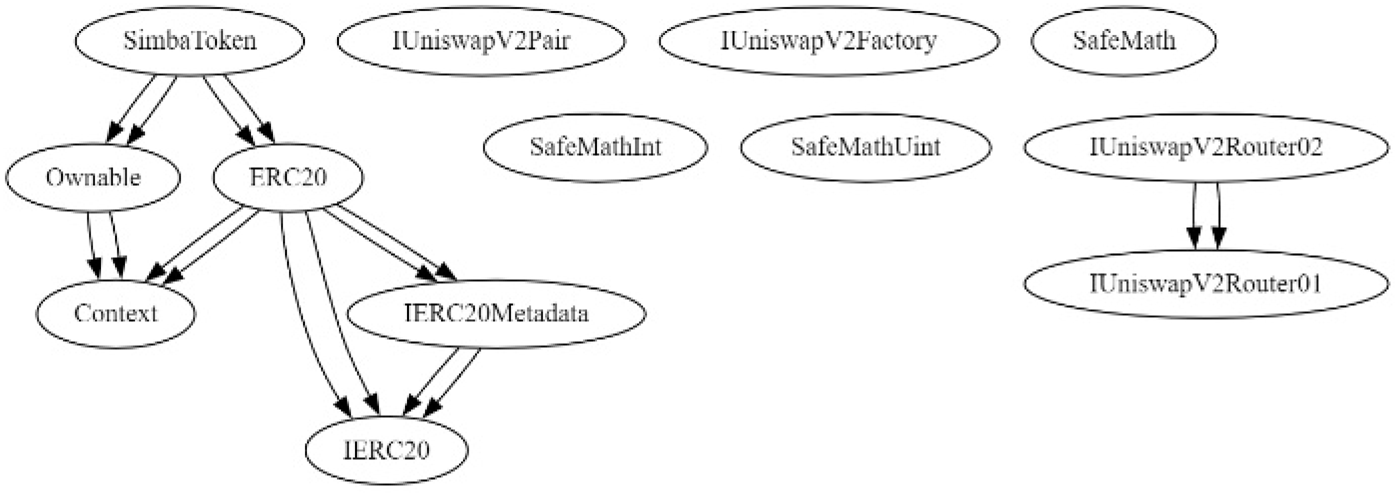
Information level is to offer suggestions for improvement of efficacity or security for features with a risk free factor.

Contract Inspection

```
|  Contract  |        Type        |       Bases      |                  |                 |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
| └ |  **Function Name**  |  **Visibility**  |  **Mutability**  |  **Modifiers**  |
| **Context** | Implementation |  |||
| **IUniswapV2Pair** | Interface |  |||
| **IUniswapV2Factory** | Interface |  |||
| **IERC20** | Interface |  |||
| **IERC20Metadata** | Interface | IERC20 |||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata |||
| **SafeMath** | Library |  |||
| **Ownable** | Implementation | Context |||
| **SafeMathInt** | Library |  |||
| **SafeMathUint** | Library |  |||
| **IUniswapV2Router01** | Interface |  |||
| **IUniswapV2Router02** | Interface | IUniswapV2Router01 |||
| **SimbaToken** | Implementation | ERC20, Ownable |||
| └ | <Constructor> | Public ❗️ | 🛑  | ERC20 |
| └ | <Receive Ether> | External ❗️ |  💵 |NO❗️ |
| └ | updateSwapTokensAtAmount | External ❗️ | 🛑  | onlyOwner |
| └ | updateSwapEnabled | External ❗️ | 🛑  | onlyOwner |
| └ | setMaxBuytx | Public ❗️ | 🛑  | onlyOwner |
| └ | setExcludeFromMaxTx | Public ❗️ | 🛑  | onlyOwner |
| └ | isExcludedFromMaxTx | Public ❗️ |   |NO❗️ |
| └ | updateBuyFees | External ❗️ | 🛑  | onlyOwner |
| └ | updateSellFees | External ❗️ | 🛑  | onlyOwner |
| └ | excludeFromFees | Public ❗️ | 🛑  | onlyOwner |
| └ | setAutomatedMarketMakerPair | Public ❗️ | 🛑  | onlyOwner |
| └ | _setAutomatedMarketMakerPair | Private 🔐 | 🛑  | |
| └ | updateMarketingWallet | External ❗️ | 🛑  | onlyOwner |
| └ | isExcludedFromFees | Public ❗️ |   |NO❗️ |
| └ | _transfer | Internal 🔒 | 🛑  | |
| └ | swapTokensForEth | Private 🔐 | 🛑  | |
| └ | addLiquidity | Private 🔐 | 🛑  | |
| └ | swapBack | Private 🔐 | 🛑  | |
```

```
Legend
|  Symbol  |  Meaning  |
|:--------:|-----------|
|    🛑    | Function can modify state |
|    💵    | Function is payable |
```

Contract Inheritance



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

Vulnerabilities Test

Test Name	Result
Function Default Visibility	Passed
Integer Overflow and Underflow	Passed
Outdated Compiler Version	Passed
Floating Pragma	Passed
Unchecked Call Return Value	Passed
Unprotected Ether Withdrawal	Passed
Unprotected SELF-DESTRUCT Instruction	Passed
Reentrancy	Passed
State Variable Default Visibility	Passed
Uninitialized Storage Pointer	Passed
Assert Violation	Passed
Use of Deprecated Solidity Functions	Passed
Delegate Call to Untrusted Callee	Passed
DoS with Failed Call	Passed
Transaction Order Dependence	Passed
Authorization through tx.origin	Passed
Block values as a proxy for time	Passed
Signature Malleability	Passed
Incorrect Constructor Name	Passed

Vulnerabilities Test

Test Name	Result
Shadowing State Variables	Passed
Weak Sources of Randomness from Chain Attributes	Passed
Missing Protection against Signature Replay Attacks	Passed
Lack of Proper Signature Verification	Passed
Requirement Violation	Passed
Write to Arbitrary Storage Location	Passed
Incorrect Inheritance Order	Passed
Insufficient Gas Griefing	Passed
Arbitrary Jump with Function Type Variable	Passed
DoS With Block Gas Limit	Passed
Typographical Error	Passed
Right-To-Left-Override control character (U+202E)	Passed
Presence of unused variables	Passed
Unexpected Ether balance	Passed
Hash Collisions With Multiple Variable Length Arguments	Passed
Message call with the hardcoded gas amount	Passed
Code With No Effects	Optimization
Unencrypted Private Data On-Chain	Passed

Findings

ID	Category	Issue	Severity
CE-01	Centralization	Max Tx with no limit	Medium
CE-OF	Centralization	Owner Accessible Functions	Optimization
GO-01	Gas Optimization	Impractical value transfer	Optimization
CS-02	Coding Standards	Using SafeMath with Solidity 0.8	Optimization

CE-01 Max Tx With No Limit

Lines # 980

```
    function setMaxBuytx(uint256 _Amount) public onlyOwner {
        maxBuyTransactionAmount = _Amount;
    }
```

Description

The above function is used to limit the amount that wallets can buy at a time. Initially set at 70% of supply, with this function it can be set to any value. Setting it to 0 will block any buy transaction. Any compromise to the owner account may allow a hacker to take advantage of this authority.

Recommendation

Set a minimum value for _Amount variable. We advise the client to carefully manage the privilege accounts' private key to avoid any potential risks of being hacked. Renounce Ownership at some point in time.

CE-OF Owner Accessible Functions

Lines # multiple lines

Description

The role OnlyOwner and authorized have authority over 11 functions that can manipulate the project functionality. Any compromise to the owner account may allow a hacker to take advantage of this authority.

Recommendation

We advise the client to carefully manage the privilege accounts' private key to avoid any potential risks of being hacked. Renounce Ownership at some point in time.

GO-01 Impractical Value Transfer

Lines # 1040

```
 if(amount == 0) {
            super._transfer(from, to, 0);
            return;
        }
```

Description

Currently, the _transfer function of the contract calls the super._transfer function when the transfer amount is equal to 0. This is unnecesary and only increases the gas cost of the function

Recommendation

It is recommended to remove this action: if(amount == 0) return;

CS-02 Using SafeMath With Solidity 0.8

Lines # multiple lines

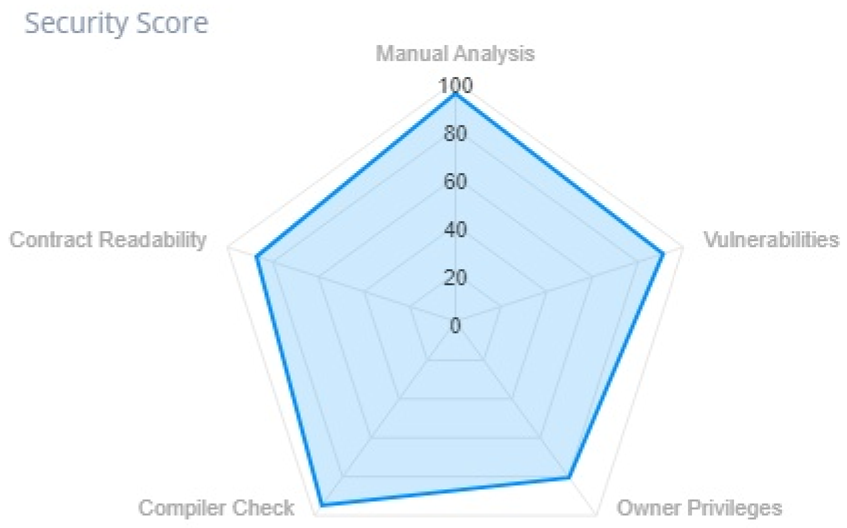
Description

SafeMath is no longer needed starting with Solidity 0.8. The compiler now has built in overflow checking. In addition, most of the functions in SafeMath, SafeMathInt and SafeMathUint library are never used and should be removed.

Recommendation

We recommend replacing Safemath operations with direct aritmetic for code readability.

Security Score



Manual Analysis Score	100
Vulnerabilities Score	96
Contract Readability Score	92
Owner Privileges	85
Compiler Score	100
Total	**94.6**

Conclusion

Simba contract uses ERC20 token standard functionality with taxes for buy/sell (up to 10%) and a limit on buy amount (medium issue). Liquidity gathered from fees is automatically added to LP.

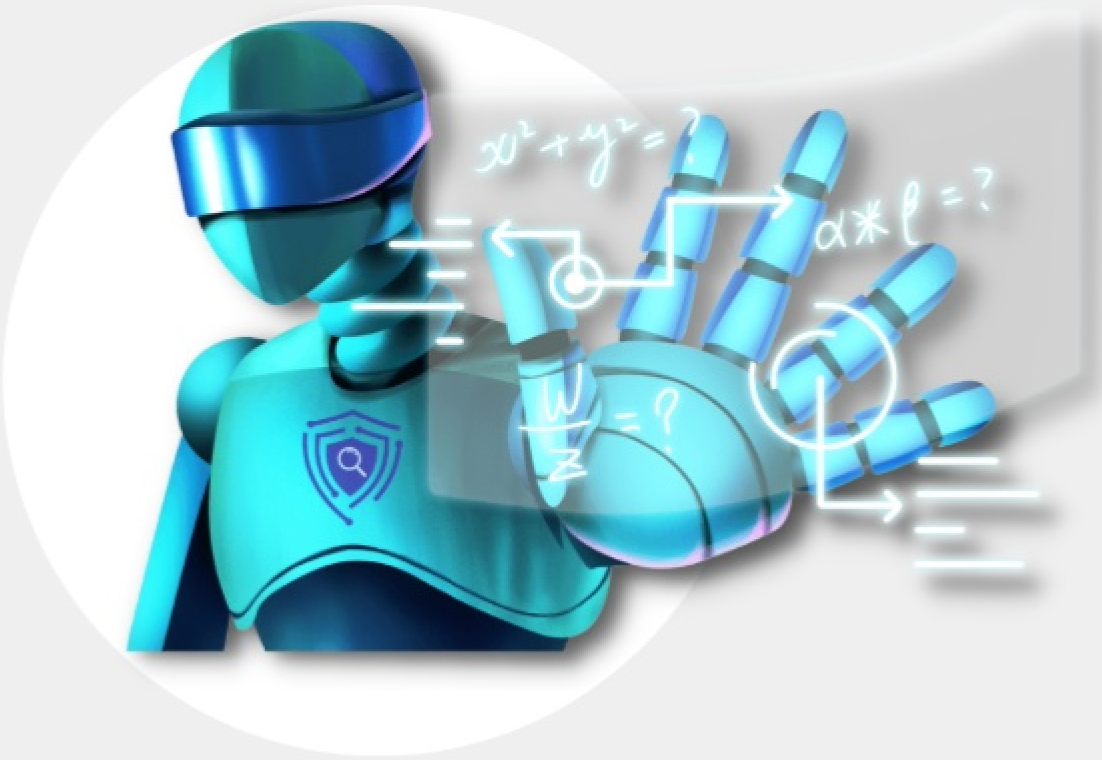
Disclaimer

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice, or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

Accuracy of Information
SafuAudit will strive to ensure the accuracy of the information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyze the on-chain smart contract source code and to provide a basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only — we recommend proceeding with several independent audits Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability or a hack. Be aware that active smart contract owner privileges constitute an elevated impact on the smart contract safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.



"Only in growth, reform, and change, paradoxically enough, is true security to be found."