Smart Contract Security Audit V2

SaferICO

https://saferico.com

06/30/2021

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

–

Table of Contents

Table of Contents **2**

Background **3**

Project Information **3**

Token Information 4

SaferICO Token Distribution: 4

Contract Interaction Details: 4

Executive Summary 5

File and Function Level Report **6**

File in Scope: **6**

Issues Checking Status **7**

Severity Definitions 8

Audit Findings 10

Conclusion **12**

Disclaimer **13**

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

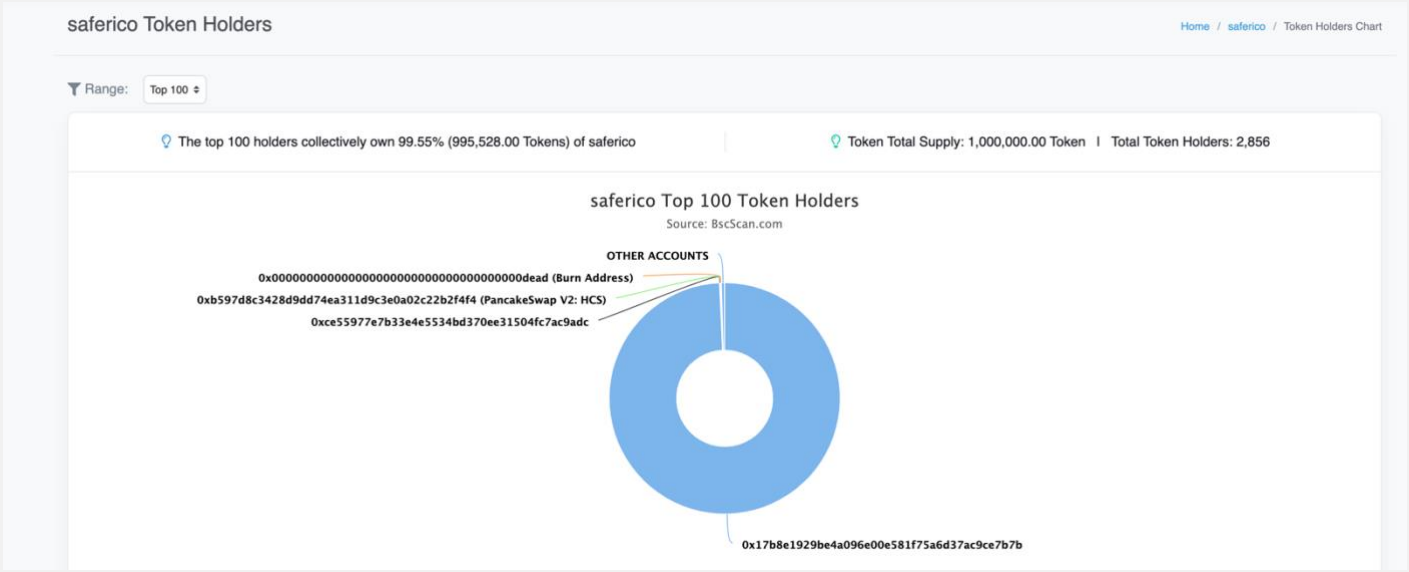
Project Information

- **Website**: https://saferico.com
- **Telegram**: https:/ t.me/SFI_ANN
- **Platform**: Binance Smart Chain
- **Contract Address**: 0xcc83c904f8754ee131577fde17dd2984109fc8aa
- SaferICO the first world token for ICO services

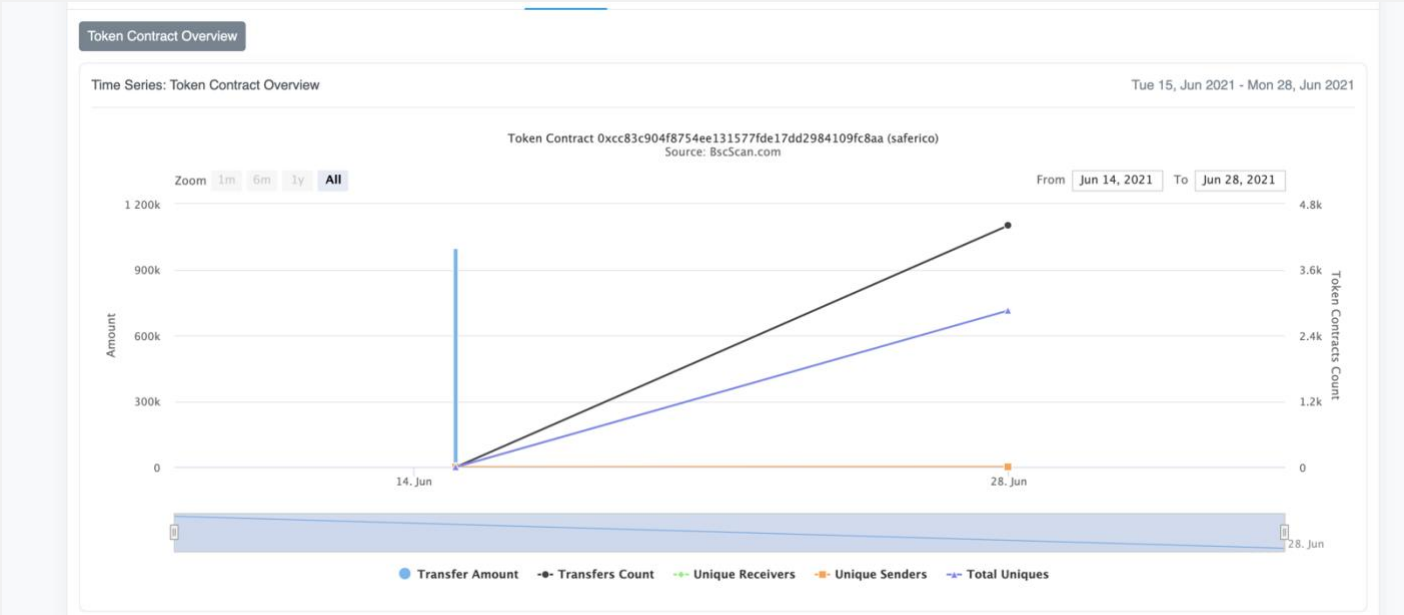
Token Information

- Name: SFI
- Total Supply: 1,000,000
- Holders: 2856 addresses
- Total transactions: 4410

SaferICO Token Distribution:



Contract Interaction Details:



Executive Summary

According to our assessment, the customer`s solidity smart contract is **Secured**.

Well Secured	
Secured	✔
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 2 low and 0 very low level issues.

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
Saferico.sol	**0x99a815166408bf52c19453b836a9dbe4a1958f13146f5fd1ad1935a89d4f615a**	0xcc83c904f8754ee131577fde17dd2984109fc8aa

- Contract: Ownable
- Inherit: Context
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type/Return Type	Score
owner	✔	Read/public	**Passed**
onlyOwner	✔	private	**Passed**
renounceOwnership	✔	Write / public	**Passed**
transferOwnership	✔	Write / public	**Passed**
geUnlockTime	✔	Read / public	**Passed**
lock	✔	Write / public	**Passed**
unlock	✔	Write / public	**Passed**

- Contract: **BEP20Token**
- Inherit: Context, IBEP20, Ownable
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
name	✔	Read / private	**Passed**
symbol	✔	Read / private	**Passed**
decimals	✔	Read / private	**Passed**
totalSupply	✔	Read / private	**Passed**
balanceOf	✔	Read / private	**Passed**
transfer	✔	Write / public	**Passed**
allowance	✔	Read/public	**Passed**
approve	✔	Write / public	**Passed**
TransferFrom	✔	Write / public	**Passed**
increaseAllowance	✔	Write / public	**Passed**
decreaseAllowance	✔	Write / public	**Passed**
multiTransfer	✔	Write / public	**Passed**
multiTransferSingleValue	✔	Write / public	**Passed**
mint	✔	Write / public	**Passed**

_approve	✔	private	**Passed**
_transfer	✔	private	**Passed**
_mint	✔	private	**Passed**
_burn	✔	private	**Passed**
_burnFrom	✔	private	**Passed**

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	**Passed**
2	Race conditions and Reentrancy. Cross-function race conditions.	**Passed**
3	Possible delays in data delivery.	**Passed**
4	Oracle calls.	**Passed**
5	Front running.	**Passed**
6	Timestamp dependence.	**Passed**
7	Integer Overflow and Underflow.	**Passed**
8	DoS with Revert.	**Passed**
9	DoS with block gas limit.	**Passed**
10	Methods execution permissions.	**Passed**
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	**Passed**
12	The impact of the exchange rate on the logic.	**Passed**
13	Private user data leaks.	**Passed**
14	Malicious Event log.	**Passed**
15	Scoping and Declarations.	**Passed**
16	Uninitialized storage pointers.	**Passed**
17	Arithmetic accuracy.	**Passed**
18	Design Logic.	**Passed**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Info	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Audit Findings

Critical:
No critical severity vulnerabilities were found.

High:
No high severity vulnerabilities were found.

Medium:
No Medium severity vulnerabilities were found.

Low:
Issue #1. Out of gas:

Approve given more allowance: -
=> I have found that in approve function user can give more allowance to a user beyond their balance.
=> It is necessary to check that user can give allowance less or equal to their amount.
=> There is no validation about user balance. So, it is good to check that a user not set approval wrongly.

- Function: - _approve
    - Here you can check that amount is not more than balance of owner.

```
function _approve(address owner, address spender, uint256 amount) internal {
    require(owner != address(0), "BEP20: approve from the zero address");
```

```
    require(spender != address(0), "BEP20: approve to the zero address");

    _allowances[owner][spender] = amount;
    emit Approval(owner, spender, amount);
  }
```

7.2: Unchecked return value or response: -

=> I have found that you are transferring fund to address using a transfer method.

=> It is always good to check the return value or response from a function call.

=> Here are some functions where you forgot to check a response.

=> I suggest, if there is a possibility then please check the response.

- Function: - clearBNB

o Here you are calling the transfer method 1 time. It is good to check that the transfer is successfully done or not.

```
function clearBNB() public onlyOwner {
    address payable _owner = msg.sender;
    _owner.transfer(address(this).balance);
  }
  function() external payable {

  }

}
```

Very Low:

No very Low severity vulnerabilities were found.

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “secured”.

- ✔ No mint function.
- ✔ No volatile code.
- ✔ No high severity issues were found.
- ✔ Contract Ownership Renounced.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Saferico hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. 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.