



Smart Contract Security Audit Report

SnowBall

November 2022

Security Status



www.hacksafe.io



Audit Details



Audited project

SnowBall



Deployer address

0x3aDE0cDAff96e7EEA78c4E8eD55427D9E5142B73



Client contacts

SnowBall Team



Blockchain

Binance smart chain



Website

Not provided

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at 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 us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: 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 TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (HackSafe) owe no duty of care towards you or any other person, nor does HackSafe 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 HackSafe 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, HackSafe hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HackSafe, 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.

Procedure

Step 1 - In-Depth Manual Review

Manual line-by-line code reviews to ensure the logic behind each function is sound and safe from various attack vectors. This is the most important and lengthy portion of the audit process (as automated tools often cannot find the nuances that lead to exploits such as flash loan attacks).

Step 2 - Automated Testing

Simulation of a variety of interactions with your Smart Contract on a test blockchain leveraging a combination of automated test tools and manual testing to determine if any security vulnerabilities exist.

Step 3 – Leadership Review

The engineers assigned to the audit will schedule meetings with our leadership team to review the contracts, any comments or findings, and ask questions to further apply adversarial thinking to discuss less common attack vectors.

Step 4 - Resolution of Issues

Consulting with the team to provide our recommendations to ensure the code's security and optimize its gas efficiency, if possible. We assist project team's in resolving any outstanding issues or implementing our recommendations.

Step 5 - Published Audit Report

Boiling down results and findings into an easy-to-read report tailored to the project. Our audit reports highlight resolved issues and any risks that exist to the project or its users, along with any remaining suggested remediation measures. Diagrams are included at the end of each report to help users understand the interactions which occur within the project.

Background

HackSafe was commissioned by SnowBall to perform an audit of smart contracts:

- <https://bscscan.com/token/0x7fdd3C80d4aE8C3B5D71aab17E13077E65037170#code>

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 understood 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.

Contract Details

Token contract details for 14.11.2022

Token Type	: Utility
Contract name	: SnowBall
Contract address	: 0x7fdd3C80d4aE8C3B5D71aab17E13077E65037170
Total supply	: 100,000,000,000
Token Ticker	: SnowB
Decimals	: 9
Token Holders	: 642
Transactions count	: 2,828
Compiler version	: v0.7.6+commit.7338295f
Contract deployer address	: 0x3aDE0cDAff96e7EEA78c4E8eD55427D9E5142B73
Owner address	: 0x3ade0cdaff96e7eea78c4e8ed55427d9e5142b73

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **“Secure”**. This token contract does contain owner control, which do not make it fully decentralized as owner does have control over smart contract.

Insecure	Poor secured	Secure	Well-secured
----------	--------------	--------	--------------

You are here 

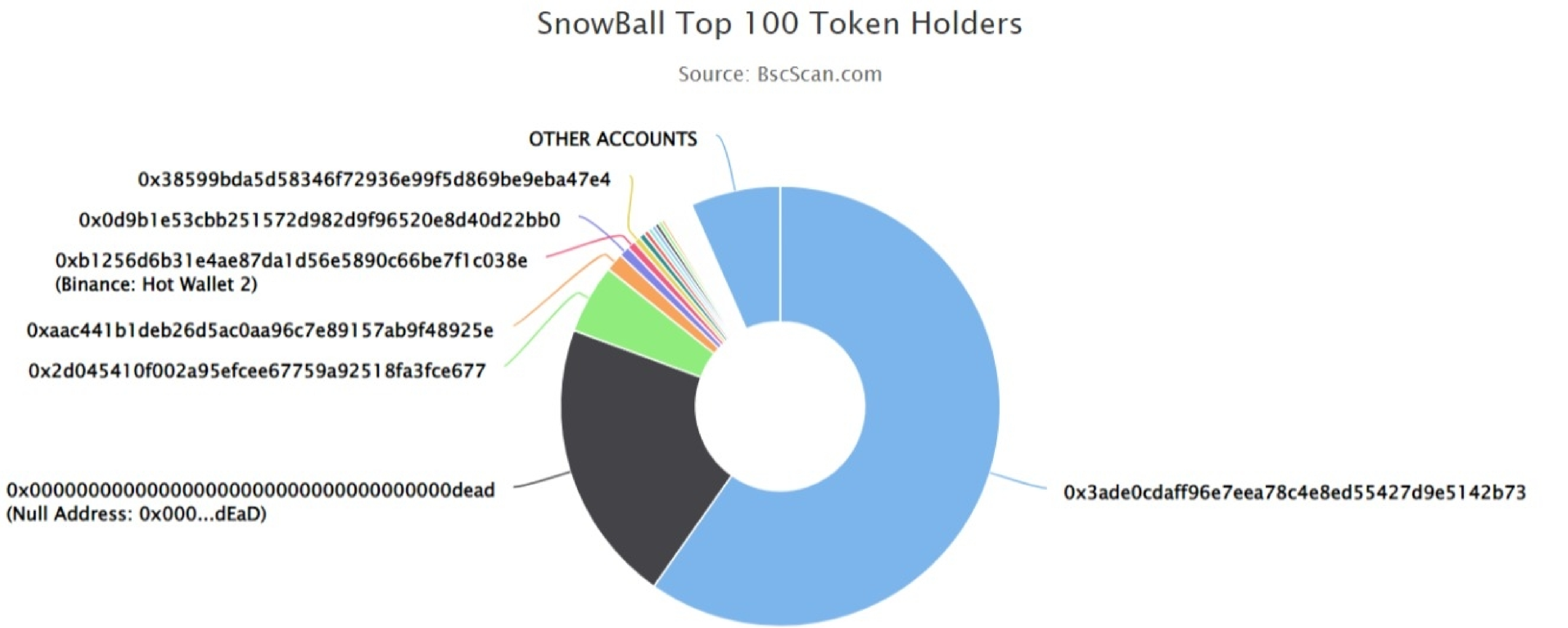
We used various tools like Slither, Mythril and Remix IDE. At the same time this finding is based on critical analysis of the manual audit. All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the issues checking status.

We found 0 critical, 0 high, 1 medium and 2 low.

SnowBall token Distribution


 The top 100 holders collectively own 93.41% (93,405,470,223.88 Tokens) of SnowBall

 Token Total Supply: 100,000,000,000.00 Token | Total Token Holders: 642



SnowBall token Top 20 Token Holders

(A total of 93,405,470,223.88 tokens held by the top 100 accounts from the total supply of 100,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0x3ade0cdaff96e7eea78c4e8ed55427d9e5142b73	59,766,282,984.034746686	59.7663%
2	Null Address: 0x000...dEaD	20,842,247,658.723792402	20.8422%
3	 0x2d045410f002a95efcee67759a92518fa3fce677	5,116,797,016.442252179	5.1168%
4	0xaac441b1deb26d5ac0aa96c7e89157ab9f48925e	1,332,924,266.826259032	1.3329%
5	0x0d9b1e53cbb251572d982d9f96520e8d40d22bb0	769,966,869.85282808	0.7700%
6	Binance: Hot Wallet 2	623,318,216.711506784	0.6233%
7	0x38599bda5d58346f72936e99f5d869be9eba47e4	481,973,836.817031452	0.4820%
8	0xa48f76284e4a8452abcd9d39a91a41001e50aefa	444,983,023.516740944	0.4450%
9	0xb5a437c79bbf2c9066cdc3c464e12b6a859d8155	349,686,261.468531927	0.3497%
10	0x39d2fd5add74bc2cd693800592680eaf6e1099aa	312,240,325.237103192	0.3122%
11	0xe8057e239a6376154adaab523e34df91c21a0259	311,511,245.988165301	0.3115%
12	0x4e84d140f32afebbec5b553be9bfbe5800b9952c	285,493,691.66087706	0.2855%
13	0xc2b0edf424b2fe23ecf5fc15b1314a998b7a9d91	263,543,724.855584598	0.2635%
14	0x8c0cb65edaa58cadcc813ccb7ec3ea93d016817c	223,563,257.253967761	0.2236%
15	0xfd3fb347dac3cf895165d2d52cc5dc36d662b192	143,071,176.587016024	0.1431%
16	0xf07c0de0a1fa32d40dcce96b0ce07faf11c0b504	133,133,004.111049486	0.1331%
17	0x152b6c9ccb5617d85b65b0cac1969bdacf0f6295	132,953,843.255891154	0.1330%
18	0x5b6b798dd88ebd2e7ec1614ad7f933996c52519f	100,447,839.691441858	0.1004%
19	0x1dc780ad7197018c0cf0e19e8c5428d6f2cfcd21	100,247,109.427848082	0.1002%
20	0x7880fc57123d9295d2980a8028d14cb313c15ba3	99,209,266.306898501	0.0992%

SnowBall token Distribution

SnowBall token Contract Overview



Contract functions details

+Context

- [Int] _msgSender
- [Int] _msgData

+ [Int] IERC20

- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] transfer
- [Ext] allowance
- [Ext] approve
- [Ext] transferFrom

+ [Lib] SafeMath

- [Int] add
- [Int] sub
- [Int] sub
- [Int] mul
- [Int] div
- [Int] div
- [Int] mod
- [Int] mod

+ [Lib] Address

- [Int] isContract
- [Int] sendValue
- [Int] functionCall
- [Int] functionCall
- [Int] functionCallWithValue
- [Int] functionCallWithValue

+Ownable (Context)

- <constructor>
- [Pub] owner
- [Pub] renounceOwnership #
-modifiers: onlyOwner
- [Pub] transferOwnership #
-modifiers: onlyOwner

Contract functions details

+SnowBall (Context, IERC20, Ownable)

- <constructor>
- [Pub] name
- [Pub] symbol
- [Pub] decimals
- [Pub] totalSupply
- [Pub] balanceOf
- [Pub] transfer #
- [Pub] allowance
- [Pub] approve #
- [Pub] transferFrom #
- [Pub] increaseAllowance #
- [Pub] decreaseAllowance #
- [Pub] isExcluded
- [Pub] totalFees
- [Pub] reflect #
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Ext] excludeAccount #
 - modifiers: onlyOwner
- [Ext] includeAccount
 - modifiers: onlyOwner
- [Pvt] _approve #
- [Pvt] _transfer #
- [Pvt] _transferStandard#
- [Pvt] _transferToExcluded #
- [Pvt] _transferFromExcluded #
- [Pvt] _transferBothExcluded #
- [Pvt] _reflectFee #
- [Pvt] _getValues
- [Pvt] _getTValues
- [Pvt] _getRValues
- [Pvt] _getRate
- [Pvt] _getCurrentSupply

(\$) = payable function

= non-constant function

Issues Checking Status

No.	Title	Status
1.	Unlocked Compiler Version	Low issue
2.	Missing Input Validation	Passed
3.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
4.	Possible delays in data delivery	Passed
5.	Oracle calls.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Medium issue
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	Private use data leaks.	Passed
13.	Malicious Event log.	Passed
14.	Scoping and Declarations.	Passed
15.	Uninitialized storage pointers.	Passed
16.	Arithmetic accuracy.	Passed
17.	Design Logic.	Passed
18.	Safe Open Zeppelin contracts implementation and usage.	Passed
19.	Incorrect Naming State Variable	Passed
20.	Too old version	Low issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a 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 assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution.

Security Issues

✔ Critical Severity Issues

No critical severity issue found.

✔ High Severity Issues

No high severity issue found.

✔ Medium Severity Issues

One medium severity issue found.

1. Out of gas limit.

- **Description**

The smart contract has functions which has used for includeAccount, _getCurrentSupply. Large length of _excluded can cause an error of out of gas for these two functions.

- **Recommendation**

It is advisable to either remove for loop or use smaller length to avoid the gas limit error while transaction.

✔ Low Severity Issues

Two low severity issue found.

1. Old compiler version

- **Description**

Contract has been deployed using too old solidity version.

- **Recommendation**

It is advisable to deploy contract using any of the latest version of solidity.

2. Unlocked Compiler Version.

- **Description**

The contract utilizes an unlocked compiler version. An unlocked compiler version in the contract's source code permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers. This can lead to ambiguity when debugging as compiler-specific bugs may occur in the codebase that would be difficult to identify over a span of multiple compiler versions rather than a specific one.

- **Recommendation**

It is advisable that the compiler version is alternatively locked at the lowest version possible so that the contract can be compiled. For example, for version ^0.7.6 the contract should contain the following line:
pragma solidity 0.7.6;

Centralization

Owner Privileges :

- Owner can transfer and renounce ownership.
- Owner can exclude and include account from fees.

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble as smart contract ownership has not been renounced. Following are Admin functions :

- Excludeaccount
- Includeaccount
- Renounceownership
- Transferownership

Conclusion

Smart contract contains low and medium severity issues! The further transfer and operations with the fund raised are not related to this particular contract.

HackSafe note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.