HackSafe

Smart Contract Security Audit Report

PenguinToken

September 2022

Security Status

★★★★☆

www.hacksafe.io

Audit Details

Audited project
PenguinToken

Deployer address
0xA239403fC6A076256E203e15122B6456C3db6f19

Client contacts
PenguinToken team

Blockchain
Avalanche

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 PenguinToken to perform an audit of smart contract:

- https://snowtrace.io/address/0xe896cdeaac9615145c0ca09c8cd5c25bced6384c#code

The purpose of the audit was to achieve the

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

The information in this report should be 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 12.09.2022

Token Type	: ERC20
Contract name	: PenguinToken
Contract address	: 0xe896CDeaAC9615145c0cA09C8Cd5C25bced6384c
Compiler version	: v0.6.12+commit.27d51765
Total upply	: 20,852,623.925949
Token ticker	: PEFI
Decimals	: 18
Token holders	: 5,949
Top 100 token	: 93.95%
Transactions count	: 1,335,328
Contract deployer address	: 0xA239403fC6A076256E203e15122B6456C3db6f19
Owner address	: 0x8ac8ed5839ba269be2619ffeb3507bab6275c257

Social profiles

Coinmarketcap profile	: https://coinmarketcap.com/currencies/penguin-finance/
Twitter profile	: https://twitter.com/penguin_defi
Coingecko profile	: https://www.coingecko.com/en/coins/penguin-finance/

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	Secure	Well-secured

You are here 

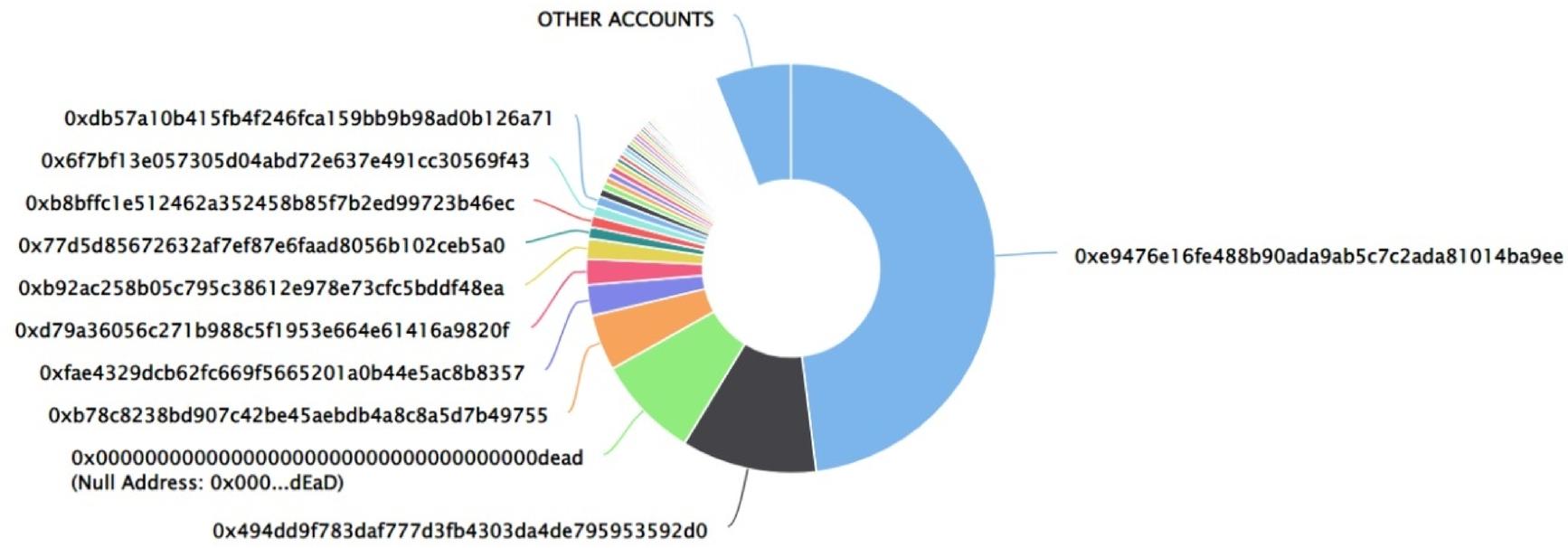
We used various tools like Slither, Mythril and Remix IDE. At the same time this finding is based on critical analysis of the manual audit. All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the issues checking status.

We found 0 critical, 0 high, 0 medium and 2 low and some very low-level issues. These issues are not critical ones.

PenguinToken Token Distribution

The top 100 holders collectively own 93.95% (19,590,063.52 Tokens) of PenguinToken

Token Total Supply: 20,852,623.93 Token | Total Token Holders: 5,949



PenguinToken Top 20 Token Holders

(A total of 19,590,063.52 tokens held by the top 100 accounts from the total supply of 20,852,623.93 token)

Rank	Address	Quantity (Token)	Percentage
1	0xe9476e16fe488b90ada9ab5c7c2ada81014ba9ee	10,020,249.205994497931940696	48.0527%
2	0x494dd9f783daf777d3fb4303da4de795953592d0	2,225,310.153344955297384628	10.6716%
3	Null Address: 0x000...dEaD	1,694,526.621401545282675415	8.1262%
4	0xb78c8238bd907c42be45aebdb4a8c8a5d7b49755	924,086.23086245747910959	4.4315%
5	0xfae4329dcb62fc669f5665201a0b44e5ac8b8357	504,967.236909574631089942	2.4216%
6	0xd79a36056c271b988c5f1953e664e61416a9820f	424,203.337792689956561276	2.0343%
7	0xb92ac258b05c795c38612e978e73cfc5bddf48ea	330,864.013653743867496739	1.5867%
8	0x77d5d85672632af7ef87e6faad8056b102ceb5a0	192,522.483525827037744786	0.9233%
9	0xb8bffc1e512462a352458b85f7b2ed99723b46ec	185,196.058658640279748397	0.8881%
10	0x6f7bf13e057305d04abd72e637e491cc30569f43	172,854.574226402462173501	0.8289%
11	0xdb57a10b415fb4f246fca159bb9b98ad0b126a71	161,650.504892536142189878	0.7752%
12	0xe03fa5b156a558f76564a5a487f9935c05d22f22	124,336.912255030674999502	0.5963%
13	0x71adb32d7c978a23e31224b8917f6595ce0a08ba	109,426.629061244073402467	0.5248%
14	0xdd3316850d2ba7b1265c9f4bf6297c42588ca0ad	102,616.069285391204671103	0.4921%
15	0xcc592739c6c64f797e46cd00f12a6f15c2df1c04	101,032.693500023944634325	0.4845%
16	0x8502ac9bfa0402be3dac2f5a8cdc955aa3a9808d	98,312.731943205942732175	0.4715%
17	0x1bb5541eccda68a352649954d4c8ece6ad68338d	89,391.528819327577542301	0.4287%
18	0x0b9753d73e1c62933e913e9c2c94f2ffa8236f6c	75,000.936856032771300188	0.3597%
19	0x1db8672f4b59e8d87aed7a083402d8243f1778cc	71,024.650161964704222672	0.3406%
20	0x9a71e0bef11444594b03cdfaf0fd53b6726e28ba	69,743.338598851151511213	0.3345%

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] tryAdd
    -[Int] trySub
    -[Int] tryMul
    -[Int] tryDiv
    -[Int] tryMod
    -[Int] add
    -[Int] sub
    -[Int] mul
    -[Int] div
    -[Int] mod
    -[Int] sub
    -[Int] div
    -[Int] mod

+ ERC20 (Context, IERC20)
    -[Pub] <constructor>
    -[Pub] name
    -[Pub] symbol
    -[Pub] decimals
    -[Pub] totalSupply
    -[Pub] balanceOf
    -[Pub] transfer #
    -[Pub] allowance
    -[Pub] approve #
    -[Pub] transferFrom #

Contract functions details

-[Pub] increaseAllowance
-[Pub] decreaseAllowance
-[Int] _transfer #
-[Int] _mint#
-[Int] _burn #
-[Int] _approve #
-[Int] _setupDecimals #
-[Int] _beforeTokenTransfer #

+Ownable (Context)
- [Int] <constructor>
-[Pub] owner
-[Pub] renounceOwnership #
-modifiers: onlyOwner
-[Pub] transferOwnership
-modifiers: onlyOwner

+ PenguinToken (ERC20, Ownable)
-[Pub] <constructor>
-[Pub] mint #
-[Ext] setMinter #
-[Ext] burnOwnTokens #
-[Ext] delegates
-[Ext] delegate
-[Ext] delegateBySig
-[Ext] getCurrentVotes
-[Ext] getPriorVotes
-[Int] _delegate
-[Int] _moveDelegates
-[Int] _writeCheckpoint
-[Int] safe32
-[Int] getChainId

($) = 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.	Passed
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
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

No medium severity issues found.

Low Severity Issues

Two low severity issue found.

1. 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

2. Too old compiler version.

- **Description**

  Contract has been deployed using too old compiler version.

- **Recommendation**

  It is advisable that the compiler version of solidity should be among the new compiler versions.

Centralization

Owner Privileges :

- PenguinToken Contract:
    - Owner can remove and transfer ownership.
    - Owner can mint new tokens.

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 functions:

- Transferownership
- Renounceownership
- Mint

Conclusion

Smart contract contains low 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.