HackSafe

Smart Contract Security Audit Report

Trivian Token

August 2022

Security Status

★★★★☆

www.hacksafe.io

Audit Details

Audited project
Trivian Token

Deployer address
0x5229c8524FcC39Ee02Dd04bA62b90B3d337B5BD8

Client contacts
Trivian Token team

Blockchain
Binance Smart chain

Website
https://trivians.io/

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 TrivianToken to perform an audit of smart contracts:

- https://bscscan.com/address/0xb465f3cb6Aba6eE375E12918387DE1eaC2301B05#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 10.08.2022

Token Type	: BEP20
Contract name	: TrivianToken
Contract address	: 0xb465f3cb6Aba6eE375E12918387DE1eaC2301B05
Compiler version	: v0.8.11+commit.d7f03943
Total supply	: 1,000,000,000
Token Ticker	: TRIVIA
Decimals	: 3
Token Holders	: 3,746
Transactions count	: 77,048
Contract deployer address	: 0x5229c8524FcC39Ee02Dd04bA62b90B3d337B5BD8
Owner address	: 0x12e7c73dff2357b1c448baa7cc5e3f4d7c9f37d9

Social profiles

Twitter Profile	: https://twitter.com/PlayTrivians
Github Profile	: https://github.com/tanselkaya/TrivianToken/commit/516294ee3f80b15827747b301d369826b676d09b
Telegram Profile	: https://t.me/TriviansGL
Coinmarketcap profile	: https://coinmarketcap.com/currencies/trivians/
Coingecko profile	: https://www.coingecko.com/en/coins/trivians/

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, 0 medium and 1 low and some very low-level issues. These issues are not critical ones.

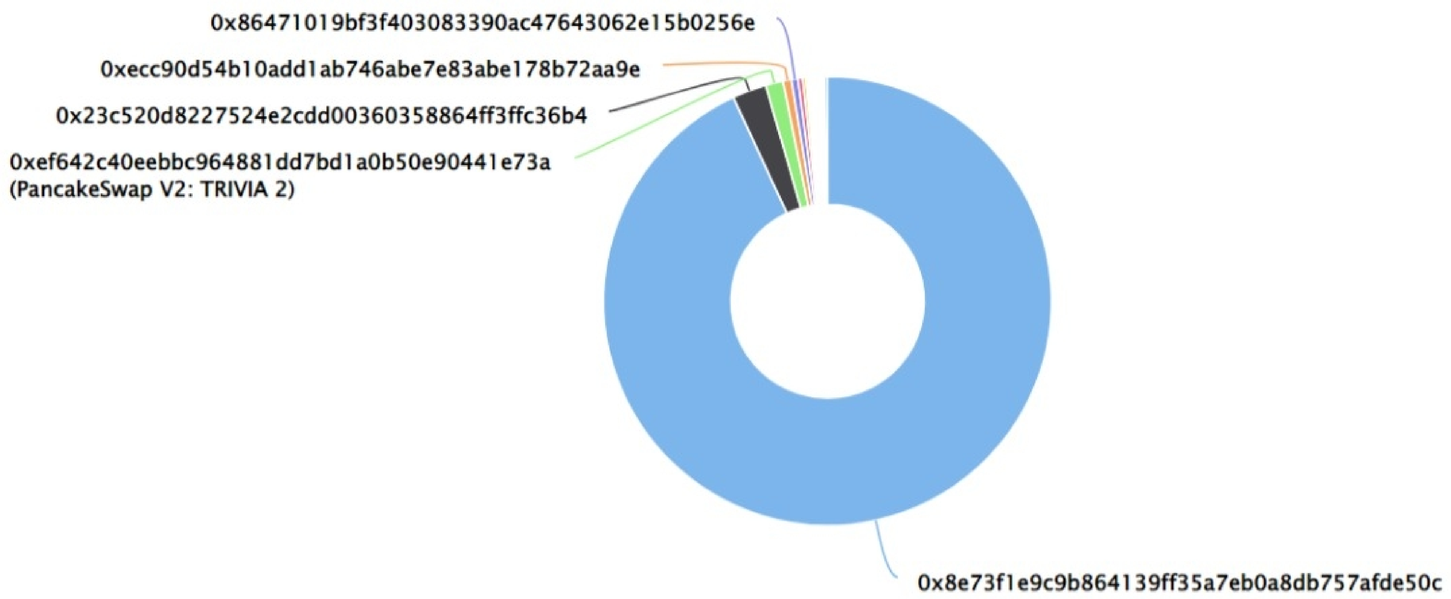
TrivianToken Token Distribution

The top 100 holders collectively own 99.78% (997,797,358.16 Tokens) of Trivian Token

Token Total Supply: 1,000,000,000.00 Token | Total Token Holders: 3,749

Trivian Token Top 100 Token Holders

Source: BscScan.com



TrivianToken Token Top 20 Token Holders

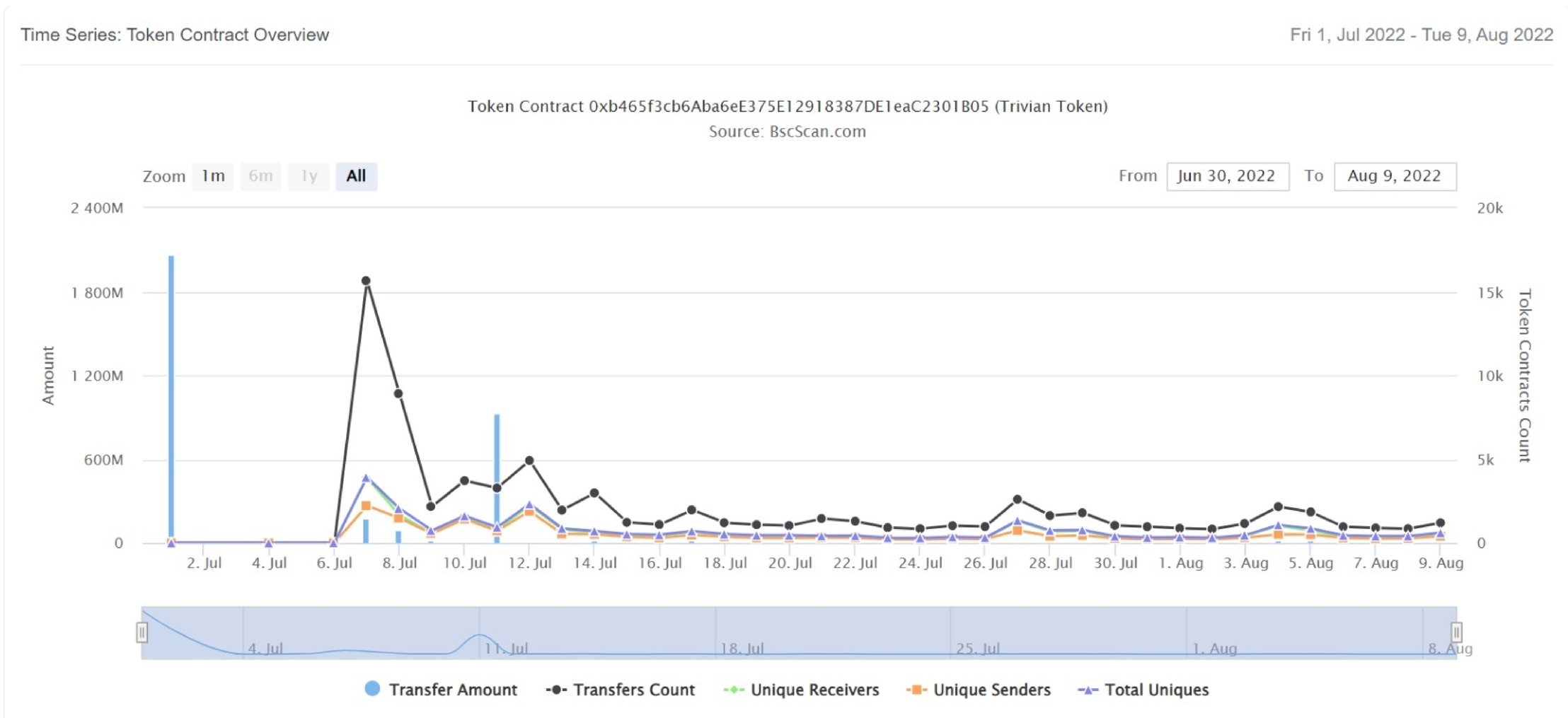
Top 1,000 holders (From a total of 3,749 holders)

First < Page 1 of 20 > Las

Rank	Address	Quantity	Percentage	Analytics
1	0x8e73f1e9c9b864139ff35a7eb0a8db757afde50c	931,080,000	93.1080%	
2	0x23c520d8227524e2cdd00360358864ff3ffc36b4	24,439,505.992	2.4440%	
3	PancakeSwap V2: TRIVIA 2	12,643,458.87	1.2643%	
4	0xecc90d54b10add1ab746abe7e83abe178b72aa9e	6,213,544.07	0.6214%	
5	0x86471019bf3f403083390ac47643062e15b0256e	4,529,654.887	0.4530%	
6	0x6fc69006fa8d09eb4be021d6916e1a160bb5ce77	3,320,000	0.3320%	
7	PancakeSwap V2: SPIN-TRIVIA	2,422,277.218	0.2422%	
8	0x0ca4d43fa7a5032a0cee1f9cbdf717dbb614df05	1,680,860	0.1681%	
9	0xcc99d88cfc9226844414469757591 8c78999de33	1,281,703.675	0.1282%	
10	0x0b8678ce34dca88aaa3ac39e01c4bac128b010b9	1,235,574.841	0.1236%	
11	0x73b764ecc70a051210529050a0d4dc3036e5827b	851,068.657	0.0851%	
12	0x9f1b0d25f450918ae1efa977ed1164ecaf69e871	691,552.973	0.0692%	
13	0x1e7d7f455daf976ca23a4d39873cf0dbb8b20150	494,626.155	0.0495%	
14	0x771e072ae5d223c616b7cefcef991f14b836175e	444,894.611	0.0445%	
15	0x3b5095a84a5902e963bf6e302fcdbc38771b0c8c	443,141.665	0.0443%	
16	0x8327b69659c5b8f445625a7b77588069fcfa9df8	431,475.077	0.0431%	
17	0x8e61d93273a2c9a1ff432f4bacd001de327dd145	403,149.96	0.0403%	
18	0xede89f69f0f654a2431e762fd5e126bcd522a82d	344,943.157	0.0345%	
19	0x11347cd6cf554002633d244d3e01a7850418a0b3	314,794.183	0.0315%	
20	0x53e06464929240c4740f480d540c6a039c1118ed	288,910.451	0.0289%	

TrivianToken Token Distribution

TrivianToken Contract Overview



Contract functions details

+ Context
    -[Int] _msgSender
    -[Int] _msgData
      -modifiers: whenPaused

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

+Pausable(Context)
    -<constructor>
    -[Pub] paused
    -[Int] _pause
      -modifiers: whenNotPaused
    -[Int] _unpause

+ [Int] IERC20
    -[Ext] totalSupply
    -[Ext] balanceOf
    -[Ext] transfer
    -[Ext] allowance
    -[Ext] approve
    -[Ext] transferFrom

+IERC20Metadata (IERC20)
    -[Ext] name
    -[Ext] symbol
    -[Ext] decimals

+ ERC20 (Context, IERC20, IERC20Metadata)
    - <constructor>
    -[Ext] getOwner
    -[Pub] name
    -[Pub] symbol
    -[Pub] decimals

Contract functions details

-[Pub] totalSupply
-[Pub] balanceOf
-[Pub] transfer #
-[Pub] allowance
-[Pub] approve #
-[Pub] transferFrom #
-[Pub] increaseAllowance
-[Pub] decreaseAllowance
-[Int] _transfer #
-[Int] _mint#
-[Int] _burn #
-[Int] _approve #
-[Int] _spendAllowance#
-[Int] _beforeTokenTransfer
-[Int] _afterTokenTransfer

+ ERC20Burnable (Context, ERC20)
-[Pub] burn #
-[Pub] burnFrom #

+ TrivianToken (ERC20, ERC20Burnable, Pausable, Ownable)
-[Ext] pause #
-modifiers: onlyOwner
-[Ext] unpause #
-modifiers: onlyOwner
-[Int] _beforeTokenTransfer #
-modifiers: whenNotPaused
-[Pub] decimals

($) = 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	Passed

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**
  No medium severity issues found.

- **Low Severity Issues**
  One 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 possible so that the contract can be compiled. For example, for version 0.8.0 the contract should contain the following line:

  pragma solidity 0.8.11;

Centralization

Owner privileges:

- HappinessToken Contract:
    - Owner can remove and transfer ownership.
    - Owner can mint tokens.
    - Owner can pause and unpause transfers.

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 but smart contract ownership has been renounced. Following are Admin functions and burner functions:

- Transferownership
- Renounceownership
- Pause
- Unpause

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.