HackSafe

Smart Contract Security Audit Report

Safe Trip Finance

November 2022

Security Status

★ ★ ★ ★ ⯪

www.hacksafe.io

Audit Details

Audited project
Safe Trip Finance

Deployer address
0x2c7E62528Fe8d55878444c76dDa05B3D298B8340

Client contacts
Safe Trip Finance 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 Safe Trip Finance to perform an audit of smart contracts:

- https://bscscan.com/token/0xe3916A4DC3C952c78348379A62d66869D9B59942#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 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 18.11.2022

Token Type	: DEFI
Contract name	: CoinToken
Contract address	: 0xe3916A4DC3C952c78348379A62d66869D9B59942
Total supply	: 50,000,000
Token Ticker	: STF
Decimals	: 18
Token Holders	: 5,664
Transactions count	: 32,460
Compiler version	: v0.4.24+commit.e67f0147
Contract deployer address	: 0x2c7E62528Fe8d55878444c76dDa05B3D298B8340
Owner address	: 0x0000000000000000000000000000000000000001

Audit Summary

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

Insecure	Poor secured	Secure	Well-secured

You are here 

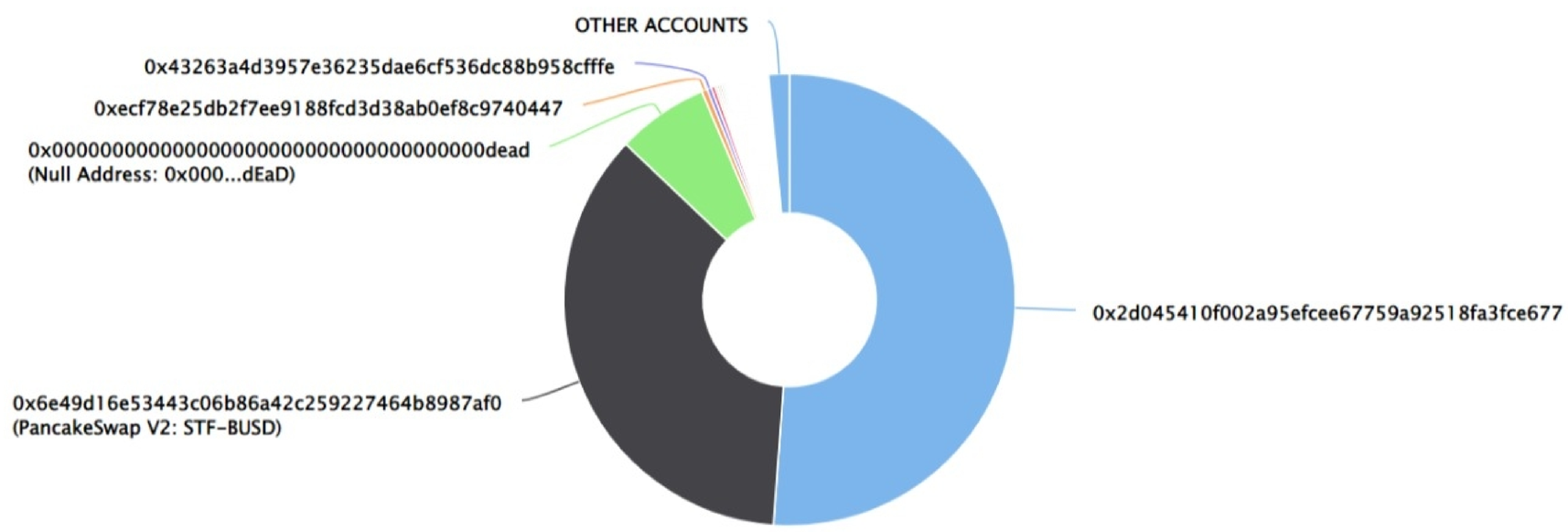
We used various tools like Slither, Mythril and Remix IDE. At the same time this finding is based on critical analysis of the manual audit. All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the issues checking status.

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

Safe Trip Finance Distribution

The top 100 holders collectively own 98.50% (49,251,262.95 Tokens) of Safe Trip Finance

Token Total Supply: 50,000,000.00 Token | Total Token Holders: 5,664



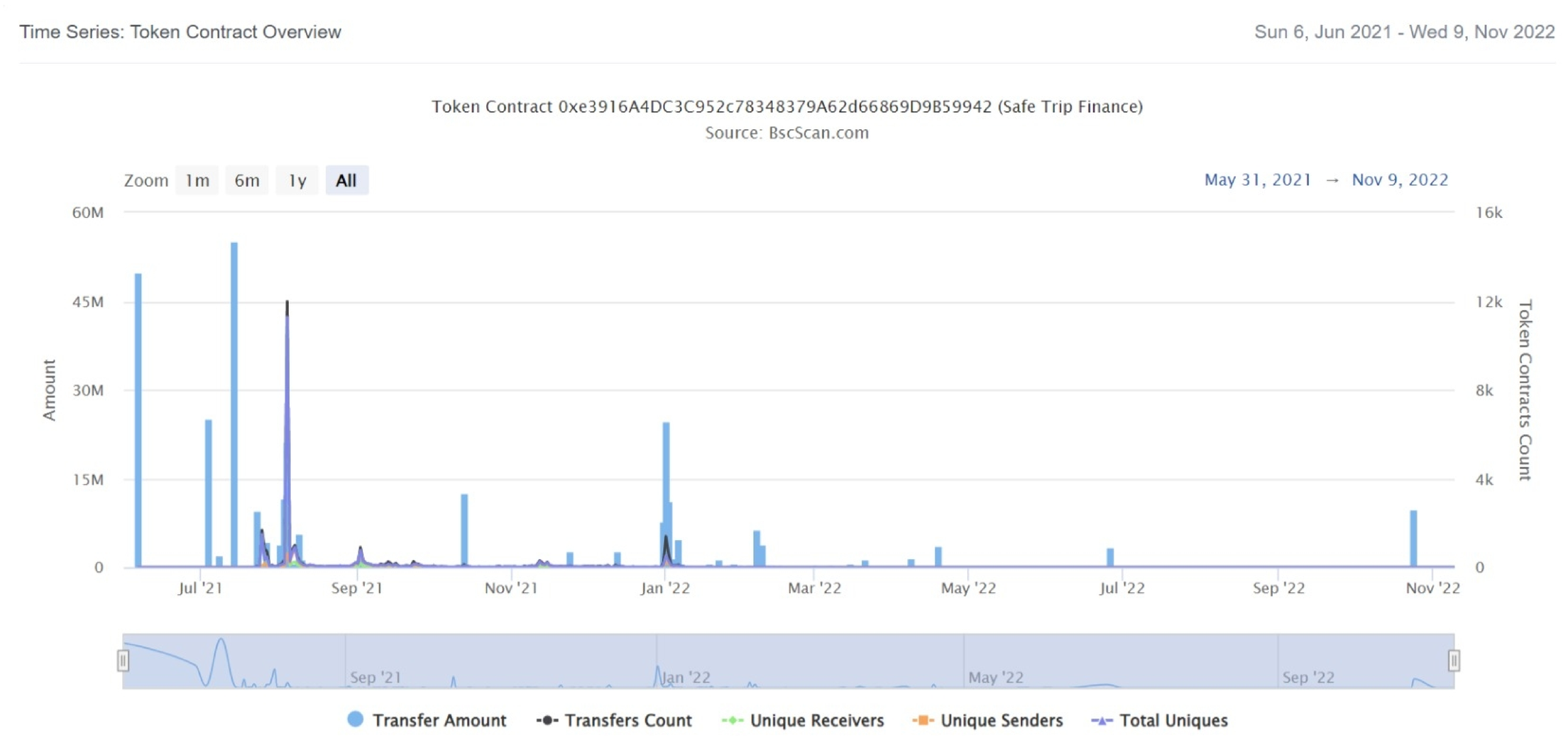
Safe Trip Finance Top 20 Token Holders

(A total of 49,251,262.95 tokens held by the top 100 accounts from the total supply of 50,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0x2d045410f002a95efcee67759a92518fa3fce677	25,576,780	51.1536%
2	PancakeSwap V2: STF-BUSD	17,964,232.564059730422107477	35.9285%
3	Null Address: 0x000...dEaD	3,279,637.734814635031354708	6.5593%
4	0xecf78e25db2f7ee9188fcd3d38ab0ef8c9740447	209,035.937831893035107907	0.4181%
5	0x43263a4d3957e36235dae6cf536dc88b958cfffe	146,695.302755145968022947	0.2934%
6	0xfd5c91b4d7404dae059617a57eec07759ba80296	138,092.29617092557761407	0.2762%
7	0x3733aaad6f7b9f52423353638e52a89ff888cddf	85,000	0.1700%
8	0x3516f344a620d685293395fa8f2b59382f5e0b05	82,600	0.1652%
9	0x9d69c4d449a72f6676b64f4bc0b4a0a9a933077d	82,323.882151507444443055	0.1646%
10	0x4b4f101672ae85fd80f90e3607ebfe9993bc0ab6	78,103.650380704670402827	0.1562%
11	0x29e8ea3f4c3f90e40b46f196d76b999bba06d8ef	67,700.000035429079786349	0.1354%
12	0xc6ef697713e95f79104ddd9f5ff9ca5a14b4cde8	65,000	0.1300%
13	0xc32f11358428a9590f5e8d7218111e2b20dd760e	62,711	0.1254%
14	0x12b43dd054f47f614e64075083bf6e3ef80ae9a7	52,325.057911056306917899	0.1047%
15	0x6cfc2be8c0f5c6990bd7fbd455b4f3c8c45f3f06	51,450.320691552613207528	0.1029%
16	0x543aefa3f7acfcce537f0397a3efdeea7c0e2df4	46,751.570069922819392125	0.0935%
17	0x2a15eb0c7ac334eb5b437fee9616e65232257227	46,067.867485578075030886	0.0921%
18	0x3cefb8333a1e822029d4e731950797ea09a8e505	45,000	0.0900%
19	0x41229d33fda2d96300ccdc2cbb1782d749e4a2f7	38,432.112204901112708732	0.0769%
20	0x00779830f04a04d32903b14601aab5e119ddc9d3	31,839.51491604337586384	0.0637%

Safe Trip Finance Distribution

Safe Trip Finance Contract Overview



Contract functions details

+[Lib] SafeMath
  -[Int] mul
  -[Int] div
  -[Int] sub
  -[Int] add

+Ownable
  -[Pub] transferOwnership #
    -modifiers: onlyOwner

+Pausable (Ownable)
  -[Pub] pause #
    -modifiers: onlyOwner, whenNotPaused
  -[Pub] unpause #
    -modifiers: onlyOwner, whenPaused

+ERC20Basic
  -[Pub] balanceOf
  -[Pub] transfer #

+ERC20 (ERC20Basic)
  -[Pub] allowance
  -[Pub] transferFrom
  -[Pub] approve

+StandardToken (ERC20)
  -[Pub] transfer #
  -[Pub] balanceOf
  -[Pub] transferFrom #
  -[Pub] approve #
  -[Pub] allowance
  -[Pub] increaseApproval #
  -[Pub] decreaseApproval #
  -[Int] _blackList #

+PausableToken (StandardToken, Pausable)
  -[Pub] transfer #
    -modifiers: whenNotPaused
  -[Pub] transferFrom #
    -modifiers: whenNotPaused

Contract functions details

-[Pub] approve #
  -modifiers: whenNotPaused
-[Pub] increaseApproval #
  -modifiers: whenNotPaused
-[Pub] decreaseApproval #
  -modifiers: whenNotPaused
-[Pub] blackListAddress #
  -modifiers: whenNotPaused, onlyOwner

+CoinToken (PausableToken)
-[Pub] <constructor> $
-[Pub] burn #
-[Int] _burn #
-[Pub] mint #
  -modifiers: onlyOwner

($) = 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 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 issue found.
- **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 ^0.4.24 the contract should contain the following line:

  pragma solidity 0.4.24;

Centralization

Owner Privileges :

- Safe Trip Finance Contract:
    - Owner can transfer ownership.
    - Owner can mint tokens.
    - Owner can pause and unpause transfers.
    - Owner can add blacklist addresses.

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:

- transferOwnership
- pause
- unpause
- blacklistAddress
- 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.