RAPID LABS

SMART CONTRACT SECURITY AUDIT

Final report **Plan: Rapid**

FastTruckCoin

April 2022

RL@RAPIDLABS.FINANCE RAPIDLABS.FINANCE

CONTENTS

INTRODUCTION

Upon the request from FastTruckCoin, Rapid Labs has prepared this report assessing the issues that exist within the contract.

Name	FastTruckCoin
Audit date	2022-04-28 — 2022-04-28
Language	Solidity
Platform	Binance Smart Chain

The reviewed version is the 9c6722f commit.

The following files are written in accordance with the OpenZeppellin library standard:

- Context.sol
- ERC20.sol
- IERC20.sol
- IERC20Metadata.sol
- Ownable.sol
- Safemath.sol

The md5 sum of the file FTCoin.sol — f10b01bc400ddbc2d9ff3076632fff2b.

The audit is created to make sure that:
There are no additional issues that have been added with the update to the original code.
Existing issues and vulnerabilities from the original code have been fixed before the contract deployment.

CONTRACTS CHECKED

FastTruck Coin is a standard ERC20 token contract.
The code can be viewed in the Github repository.

Name	Address
FTCoin.sol	https://github.com/FastTruckUs/FastTruckCoin/blob/9c6722f8d4f4b06147e655af7166c608f7a73546/FTCoin.sol

PROCEDURE

Our audits follow this procedure:

Automated analysis

- Analysis with the involvement of available automated Solidity analysis tools.
- Manual confirmation of the issues discovered by the tool.

Manual audit

- Analysis of the code performed manually by auditors
- A thorough check of Smart contract's logic

ATTACKS CHECKED

Title	Check result
Unencrypted Private Data On-Chain	✓ passed
Code With No Effects	✓ passed
Message call with hardcoded gas amount	✓ passed
Typographical Error	✓ passed
DoS With Block Gas Limit	✓ passed
Presence of unused variables	✓ passed
Incorrect Inheritance Order	✓ passed

Title	Check result
Requirement Violation	✓ passed
Weak Sources of Randomness from Chain Attributes	✓ passed
Shadowing State Variables	✓ passed
Incorrect Constructor Name	✓ passed
Block values as a proxy for time	✓ passed
Authorization through tx.origin	✓ passed
DoS with Failed Call	✓ passed
Delegatecall to Untrusted Callee	✓ passed
Use of Deprecated Solidity Functions	✓ passed
Assert Violation	✓ passed
State Variable Default Visibility	✓ passed
Reentrancy	✓ passed
Unprotected SELFDESTRUCT Instruction	✓ passed
Unprotected Ether Withdrawal	✓ passed

Title	Check result
Unchecked Call Return Value	✓ passed
Floating Pragma	✗ not passed
Outdated Compiler Version	✓ passed
Integer Overflow and Underflow	✓ passed
Function Default Visibility	✓ passed

CLASSIFICATION OF ISSUES

Rapid Labs team separates issues into 3 types: High, Average, and Low significance.

High significance indicates an immediate need for attention. An issue of this type can be the reason for the loss of funds or contradictions in the contract's logic.

Average significance describes an issue that can potentially lead to issues within a contract or its complete failure. Though not as severe as the first type, these issues still require fixing.

Low significance indicates an issue that doesn't affect the contract but should be fixed or considered.

ISSUES

High significance

1. Non-regulated Fee changes (FTCoin)
Two functions: setBuyFee() and setSellFee() don't regulate fee changes. This, in case the owner is compromised, can potentially lead to a failure of the contract.
If, for instance, buyBurnTax is set to 110, a permanent failure of the contract will occur in 142L.

Recommended action:
Restricting the inputs in the setBuyFee() and setSellFee() functions via require().
Adding a check that _fee falls within a specific range of values (for example, from 5 to 20).

Average significance

1. 0 values not passed to transfer() (FTCoin)
ERC20 standard(link) indicated that tokens must be able to pass a 0 value in the transfer() and transferFrom() functions.

Recommended action:
Adding handling of the 0 value to the function transfer().

Low significance

1. Setting a public modifier for the variables (FTCoin)
There should be a public modifier set for the variables 20-22L.

Recommended action:
Adding a public modifier for these variables to increase the transparency for users.

2. Lack of events (FTCoin)
Several functions in contract lack corresponding events:
1) setBuyFee()
2) setSellFee()
3) transfer()

Recommended action:
Creating events for the listed functions to provide transparency.

3. Non-fixed Pragma (FTCoin)
Deploy contracts with the compiler version and flags used during testing. To avoid deploying with an outdated compiler and accidentally triggering bugs, lock the pragma.

Recommended action:
Locking the pragma version;
Review the known bugs (link) for the selected compiler version.

4. Following Naming convention (FTCoin)
Naming for the constants is preferred in all capital letters with underscores used for separating words(link).
The following constants are not named according to the convention:
deadWallet

Recommended action:
Changing the name of the deadWallet constant to improve the quality and readability of the code.

5. Unnecessary Safemath library (FTCoin)
In Solidity 0.8 and later, the check for overflow/underflow is implemented on the language level - the validation is added to the bytecode during compilation.
SafeMath library isn't needed for Solidity 0.8 and later (link).
If it's used, additional gas is consumed during deployment and several calls.

Recommended action:
Removing interactions with the SafeMath library.

CONCLUSION

In the audited FastTruckCoin FTCoin contract, the Rapid Labs team has discovered 1 high significance, 1 average, and 5 low significance issues.

DISCLAIMER

The goal of this report is to provide the quality assessment of the code to the clients initially requesting the services from Rapid Labs.

This report exists under the Terms and Conditions described in the Services Agreement.
This report is provided in accordance with the Services described in said Agreement.
The Company will only use this report under the circumstances covered in the Services Agreement.

This report should not be transmitted, disclosed, referred to, or relied upon by anyone for any purpose, unless they acquire written permission to do so from Rapid Labs.

This report is neither endorsing nor criticizing any project, company, or party involved in the making of the report. This report should not be viewed as a financial recommendation or used as a guide for investment or business propositions. This document does not assess the financial or economical value of the project, it does not guarantee satisfactory results to anyone involved with the project reviewed.

SECURITY ANALYSIS OUTCOME

```
FTCoin.setBuyFee(uint16) (contracts/FTCoin.sol#88-90) should emit an event for:
        - buyBurnTax = _fee (contracts/FTCoin.sol#89)
FTCoin.setSellFee(uint16) (contracts/FTCoin.sol#92-94) should emit an event for:
        - sellBurnTax = _fee (contracts/FTCoin.sol#93)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-
events-arithmetic

FTCoin.updateUniswapV2Router(address)._uniswapV2Pair (contracts/FTCoin.sol#66-67)
lacks a zero-check on :
                - uniswapV2Pair = _uniswapV2Pair (contracts/FTCoin.sol#68)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-
zero-address-validation

Reentrancy in FTCoin.constructor() (contracts/FTCoin.sol#37-57):
        External calls:
        - _uniswapV2Pair =
IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Ro
uter.WETH()) (contracts/FTCoin.sol#42-43)
        State variables written after the call(s):
        - _mint(owner(),3500000 * 10 ** uint256(decimals())) (contracts/
FTCoin.sol#56)
                - _balances[account] += amount (contracts/ERC20.sol#299)
        - excludeFromFees(owner(),true) (contracts/FTCoin.sol#53)
                - _isExcludedFromFees[account] = excluded (contracts/FTCoin.sol#72)
        - excludeFromFees(address(this),true) (contracts/FTCoin.sol#54)
                - _isExcludedFromFees[account] = excluded (contracts/FTCoin.sol#72)
        - _mint(owner(),3500000 * 10 ** uint256(decimals())) (contracts/
FTCoin.sol#56)
                - _totalSupply += amount (contracts/ERC20.sol#298)
        - _setAutomatedMarketMakerPair(_uniswapV2Pair,true) (contracts/
FTCoin.sol#51)
                - automatedMarketMakerPairs[pair] = value (contracts/
FTCoin.sol#113)
        - buyBurnTax = 5 (contracts/FTCoin.sol#48)
        - sellBurnTax = 10 (contracts/FTCoin.sol#49)
        - uniswapV2Pair = _uniswapV2Pair (contracts/FTCoin.sol#46)
        - uniswapV2Router = _uniswapV2Router (contracts/FTCoin.sol#45)
Reentrancy in FTCoin.updateUniswapV2Router(address) (contracts/FTCoin.sol#59-69):
        External calls:
        - _uniswapV2Pair =
IUniswapV2Factory(uniswapV2Router.factory()).createPair(address(this),uniswapV2Rout
er.WETH()) (contracts/FTCoin.sol#66-67)
        State variables written after the call(s):
        - uniswapV2Pair = _uniswapV2Pair (contracts/FTCoin.sol#68)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-2
```

```
Reentrancy in FTCoin.constructor() (contracts/FTCoin.sol#37-57):
        External calls:
        - _uniswapV2Pair =
IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Ro
uter.WETH()) (contracts/FTCoin.sol#42-43)
        Event emitted after the call(s):
        - ExcludeFromFees(account,excluded) (contracts/FTCoin.sol#74)
                - excludeFromFees(address(this),true) (contracts/FTCoin.sol#54)
        - ExcludeFromFees(account,excluded) (contracts/FTCoin.sol#74)
                - excludeFromFees(owner(),true) (contracts/FTCoin.sol#53)
        - SetAutomatedMarketMakerPair(pair,value) (contracts/FTCoin.sol#115)
                - _setAutomatedMarketMakerPair(_uniswapV2Pair,true) (contracts/
FTCoin.sol#51)
        - Transfer(address(0),account,amount) (contracts/ERC20.sol#300)
                - _mint(owner(),3500000 * 10 ** uint256(decimals())) (contracts/
FTCoin.sol#56)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-3

Different versions of Solidity is used:
        - Version used: ['>=0.5.0', '>=0.6.2', '^0.8.0', '^0.8.6']
        - ^0.8.0 (contracts/Context.sol#3)
        - ^0.8.0 (contracts/ERC20.sol#3)
        - ^0.8.6 (contracts/FTCoin.sol#9)
        - ^0.8.0 (contracts/IERC20.sol#3)
        - ^0.8.0 (contracts/IERC20Metadata.sol#3)
        - >=0.5.0 (contracts/IUniswapV2Factory.sol#5)
        - >=0.6.2 (contracts/IUniswapV2Router01.sol#5)
        - >=0.6.2 (contracts/IUniswapV2Router02.sol#5)
        - ^0.8.0 (contracts/Ownable.sol#3)
        - ^0.8.0 (contracts/SafeMath.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used

Context._msgData() (contracts/Context.sol#20-22) is never used and should be
removed
ERC20._burn(address,uint256) (contracts/ERC20.sol#316-331) is never used and should
be removed
SafeMath.add(uint256,uint256) (contracts/SafeMath.sol#111-113) is never used and
should be removed
SafeMath.div(uint256,uint256,string) (contracts/SafeMath.sol#209-218) is never used
and should be removed
SafeMath.mod(uint256,uint256) (contracts/SafeMath.sol#169-171) is never used and
should be removed
SafeMath.mod(uint256,uint256,string) (contracts/SafeMath.sol#235-244) is never used
and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/SafeMath.sol#186-195) is never used
and should be removed
SafeMath.tryAdd(uint256,uint256) (contracts/SafeMath.sol#20-30) is never used and
should be removed
SafeMath.tryDiv(uint256,uint256) (contracts/SafeMath.sol#74-83) is never used and
should be removed
```

```
SafeMath.tryMod(uint256,uint256) (contracts/SafeMath.sol#90-99) is never used and
should be removed
SafeMath.tryMul(uint256,uint256) (contracts/SafeMath.sol#53-67) is never used and
should be removed
SafeMath.trySub(uint256,uint256) (contracts/SafeMath.sol#37-46) is never used and
should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (contracts/Context.sol#3) allows old versions
Pragma version^0.8.0 (contracts/ERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (contracts/IERC20Metadata.sol#3) allows old versions
Pragma version>=0.5.0 (contracts/IUniswapV2Factory.sol#5) allows old versions
Pragma version>=0.6.2 (contracts/IUniswapV2Router01.sol#5) allows old versions
Pragma version>=0.6.2 (contracts/IUniswapV2Router02.sol#5) allows old versions
Pragma version^0.8.0 (contracts/Ownable.sol#3) allows old versions
Pragma version^0.8.0 (contracts/SafeMath.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Parameter FTCoin.setBuyFee(uint16)._fee (contracts/FTCoin.sol#88) is not in
mixedCase
Parameter FTCoin.setSellFee(uint16)._fee (contracts/FTCoin.sol#92) is not in
mixedCase
Constant FTCoin.deadWallet (contracts/FTCoin.sol#17-18) is not in
UPPER_CASE_WITH_UNDERSCORES
Function IUniswapV2Router01.WETH() (contracts/IUniswapV2Router01.sol#10) is not in
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#conformance-to-solidity-naming-conventions

Variable
IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,add
ress,uint256).amountADesired (contracts/IUniswapV2Router01.sol#15) is too similar
to
IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,add
ress,uint256).amountBDesired (contracts/IUniswapV2Router01.sol#16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-
names-are-too-similar

FTCoin.constructor() (contracts/FTCoin.sol#37-57) uses literals with too many
digits:
        - _mint(owner(),3500000 * 10 ** uint256(decimals())) (contracts/
FTCoin.sol#56)
FTCoin.slitherConstructorConstantVariables() (contracts/FTCoin.sol#11-159) uses
literals with too many digits:
        - deadWallet = address(0x000000000000000000000000000000000000dEaD)
(contracts/FTCoin.sol#17-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits
```

```
name() should be declared external:
        - ERC20.name() (contracts/ERC20.sol#60-62)
symbol() should be declared external:
        - ERC20.symbol() (contracts/ERC20.sol#68-70)
totalSupply() should be declared external:
        - ERC20.totalSupply() (contracts/ERC20.sol#92-94)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (contracts/ERC20.sol#99-107)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (contracts/ERC20.sol#117-125)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (contracts/ERC20.sol#130-138)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (contracts/ERC20.sol#147-155)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (contracts/ERC20.sol#170-187)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (contracts/ERC20.sol#201-212)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (contracts/ERC20.sol#228-243)
excludeMultipleAccountsFromFees(address[],bool) should be declared external:
        - FTCoin.excludeMultipleAccountsFromFees(address[],bool) (contracts/
FTCoin.sol#77-86)
setAutomatedMarketMakerPair(address,bool) should be declared external:
        - FTCoin.setAutomatedMarketMakerPair(address,bool) (contracts/
FTCoin.sol#96-106)
isExcludedFromFees(address) should be declared external:
        - FTCoin.isExcludedFromFees(address) (contracts/FTCoin.sol#118-120)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/Ownable.sol#56-58)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (contracts/Ownable.sol#64-70)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
. analyzed (10 contracts with 77 detectors), 50 result(s) found
```

RAPID LABS

YOUR PROJECT SECURED

RAPIDLABS.FINANCE

RL@RAPIDLABS.FINANCE