KISHIELD

Security Audit

FunStep Token

May 12, 2022

FUNSTEP

Table of Contents

Audit Summary

This report has been prepared for FunStep Token on the Binance Chain network. KISHIELD provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.

The auditing process pays special attention to the following considerations:

- Ensuring contract logic meets the specifications and intentions of the client without exposing the user's funds to risk.
- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Project Overview

Token Summary

Parameter	Result
Address	0x85a513A3CBb5bdd9A332CF8b9Dd9D8F95f69A3eA
Name	FunStep
Token Tracker	FunStep (FST)
Decimals	9
Supply	50,000,000
Platform	Binance Chain
compiler	v0.8.7+commit.e28d00a7
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/ address/0x85a513A3CBb5bdd9A332CF8b9Dd9D8F95f69A3eA
Url	https://www.funstep.app

Main Contract Assessed

Name	Contract	Live
FunStep	0x85a513A3CBb5bdd9A332CF8b9Dd9D8F95f69A3eA	Yes

Smart Contract Vulnerability Checks

Vulnerability	Automatic Scan	Manual Scan	Result
Unencrypted Private Data On-Chain	Complete	Complete	✓ Low / No Risk
Code With No Effects	Complete	Complete	✓ Low / No Risk
Message call with hardcoded gas amount	Complete	Complete	✓ Low / No Risk
Hash Collisions With Multiple Variable Length Arguments	Complete	Complete	✓ Low / No Risk
Unexpected Ether balance	Complete	Complete	✓ Low / No Risk
Presence of unused variables	Complete	Complete	✓ Low / No Risk
Right-To-Left-Override control character (U+202E)	Complete	Complete	✓ Low / No Risk
Typographical Error	Complete	Complete	✓ Low / No Risk
DoS With Block Gas Limit	Complete	Complete	✓ Low / No Risk
Arbitrary Jump with Function Type Variable	Complete	Complete	✓ Low / No Risk
Insufficient Gas Griefing	Complete	Complete	✓ Low / No Risk
Incorrect Inheritance Order	Complete	Complete	✓ Low / No Risk
Write to Arbitrary Storage Location	Complete	Complete	✓ Low / No Risk
Requirement Violation	Complete	Complete	✓ Low / No Risk
Missing Protection against Signature Replay Attacks	Complete	Complete	✓ Low / No Risk
Weak Sources of Randomness from Chain Attributes	Complete	Complete	✓ Low / No Risk

Vulnerability	Automatic Scan	Manual Scan	Result
Authorization through tx.origin	Complete	Complete	✓ Low / No Risk
Delegatecall to Untrusted Callee	Complete	Complete	✓ Low / No Risk
Use of Deprecated Solidity Functions	Complete	Complete	✓ Low / No Risk
Assert Violation	Complete	Complete	✓ Low / No Risk
Reentrancy	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
Unprotected Ether Withdrawal	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Value	Complete	Complete	✓ Low / No Risk
Outdated Compiler Version	Complete	Complete	✓ Low / No Risk
Integer Overflow and Underflow	Complete	Complete	✓ Low / No Risk
Function Default Visibility	Complete	Complete	✓ Low / No Risk

Contract Ownership

The contract ownership of FunStep is not currently renounced. The ownership of the contract grants special powers to the protocol creators, making them the sole addresses that can call sensible ownable functions that may alter the state of the protocol.

The current owner is the address 0x83e912370e83D508097Bf341b623BE0980EA32f1 which can be viewed from:
HERE

The owner wallet has the power to call the functions displayed on the priviliged functions chart below, if the owner wallet is compromised this privileges could be exploited.

We recommend the team to renounce ownership at the right timing if possible, or gradually migrate to a timelock with governing functionalities in respect of transparency and safety considerations.

Important Notes To The Users:

- The owner cannot mint tokens after intial deployment.
- Once the owner renounces ownership of the contract, none of the following are applicable.
- The owner can change the max tx amount addressed as _maxOnceEat in the contract with no restrictions.
- The owner can move tokens from any wallet without permission via the multiTransfer_fixed function.
- The owner can stop users from selling by adding them to the whoCantEat mapping.
- The owner can set the whenEat and cantEatCake variables, if they are set to large values users will be added to the whoCantEat mapping and will not be able to sell.
- The owner can stop non-isExcludedFromCut users from selling by setting nowIsEatingCake to false.
- The marketing wallet is addressed as doYouLikeBase and the team wallet as inTheMTFFace.
- The owner can enable/disable the auto liquidity mechanism.
- The owner can change the max wallet amount addressed as _maxTotalEat in the contract with no restrictions.
- The owner can set the buy/sell fees to 100%.
- The owner can add/remove addresses from the whoCantEat mapping.
- The owner can change distribution settings with no restrictions.
- High-risk Exploits/Vulnerabilities Were Found in token Source Code.

Including access to user funds and stopping users from trading.

Audit Not Passed

Technical Findings Summary

Classification of Issues

Severity	Description
🔴 High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency
🟠 Medium	Bugs or issues with that may be subject to exploit, though their impact is somewhat limited.Issues under this classification are recommended to be fixed as soon as possible.
🟡 Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
🟣 Info	Consistency, syntax or style best practices. Generally pose a negligible level of risk, if any.

Findings

Severity	Found
🔴 High	2
🟠 Medium	0
🟡 Low	0
🟣 Info	4
Total	6

Findings

Users fund access

ID	Severity	Contract	Function
01	🔴 High	FunStep	Function multiTransfer_fixed()

Description

The owner can move funds from any wallet to n-wallets without the user permission. _basicTransfer does not check for token allowances and the owner can input any address in the 'from' field.

Recommendation

We recommend checking the allowances of the 'from' wallet or using _transfer() instead of _basicTransfer()

Tax too high

ID	Severity	Contract	Function
02	🔴 High	FunStep	Function setBBB() & setSSS()

Description

The owner can change the fees to 100% with no restrictions at any time. If the owner changes it to a high value there will be 100% tax forever for non-tax-exempt users.

Recommendation

We recommend adding a require statement to limit the max fee amount.

Variables could be declared as constant

ID	Severity	Contract	Function
03	● Informational	FunStep	variable deadAddress

Description

Gas Optimization. Variables that are never changed could be declared as constant.

Recommendation

We recommend declaring those variables as constant.

Public function that could be declared external

ID	Severity	Contract	Function
04	● Informational	FunStep	Functions setWhoCantEat, manage_CantEat, isCantEat, manageExcludeFromCut, setisExcludedFromCut, letsEatCake

Description

Gas Optimization. Public function that could be declared external

Recommendation

Public functions that are never called by the contract should be declared external to save gas.

Missing events arithmetic

ID	Severity	Contract	Function
05	● Informational	FunStep	Missing events for letsEatCake, setBBB, setSSS, setDistributionSettings, setMaxOnceEat, setMaxTotalEat, setNumTokensBeforeSwap

Description

Functions that change critical arithmetic parameters should emit an event.

Recommendation

Emit corresponding events for critical parameter changes.

Division before Multiplication

ID	Severity	Contract	Function
06	● Informational	FunStep	function _transfer()

Description

Precision Loss. '_sellReserveFeeAmount = amount.div(100).mul(1)'. Division before multiplication can result in truncation and less accurate results

Recommendation

Multiplication should be performed before division to not lose precision.

Priviliged Functions (onlyOwner)

Function Name	Parameters	Visibility
waiveOwnership	none	public
transferOwnership	address newOwner	public
letsEatCake	uint256 canteat, bool status	public
setMarketPairStatus	address account, bool newValue	public
setisOnceEatExempt	address holder, bool exempt	external
setisExcludedFromCut	address account, bool newValue	public
manageExcludeFromCut	calldata addresses, bool status	public
setBBB	uint256 newLiquidityTax, uint256 newMarketingTax, uint256 newTeamTax	external
setSSS	uint256 newLiquidityTax, uint256 newMarketingTax, uint256 newTeamTax	external
setDistributionSettings	uint256 newLiquidityShare, uint256 newMarketingShare, uint256 newTeamShare	external
setMaxOnceEat	uint256 newMaxOnceEat	external
enableMaxEat	bool newValue	external
setisMaxEatExempt	address holder, bool exempt	external
setMaxTotalEat	uint256 newMaxTotalEat	external
setNumTokensBeforeSwap	uint256 newValue	external

Function Name	Parameters	Visibility
setdoYouLikeBase	address newAddress	external
setinTheMTFFace	address newAddress	external
setSwapAndLiquifyEnabled	bool _enabled	public
setSwapAndLiquifyBySmallOnly	bool newValue	public
multiTransfer_fixed	address from, calldata addresses, uint256 amount	external
manage_CantEat	calldata addresses, bool status	public
setWhoCantEat	address recipient, bool status	public
swapAndLiquify	none	private

Statistics

Liquidity Info

Parameter	Result
Pair Address	0x9f12987F13447f918e0C298877d3Fc894e6b77Fa
FST Reserves	0.00 FST
BNB Reserves	0.00 BNB
Liquidity Value	$0 USD

Token (FST) Holders Info

Parameter	Result
FST Percentage Burnt	0.00%
FST Amount Burnt	0 FST
Top 10 Percentage Own	100.00%
Top 10 Amount Owned	50,000,000 FST
Top 10 Aprox Value	$NaN USD

LP (FST/BNB) Holders Info

Parameter	Result
FST/BNB % Burnt	0.00%
FST/BNB Amount Burnt	0 FST
Top 10 Percentage Owned	0.00%
Top 10 Amount Owned	0 FST
Locked Tokens Percentage	0.00%
Locked Tokens Amount	0 FST

* All the data diplayed above was taken on-chain at block 17724645
* The tokens on industry-standard burn wallets are not included on the top 10 wallets calculations

Liquidity Ownership

The token does not have liquidity at the moment of the audit, block 17724645

KISHIELD

Disclaimer

KISHIELD has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report. Information is provided ‘as is’, and KISHIELD is under no covenant to the completeness, accuracy or solidity of the contracts audited. In no event will KISHIELD or its partners, employees, agents or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

The assessment services provided by KISHIELD is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.