Blockchain Security - Smart Contract Audits

Security Assessment

January 14, 2022

Disclaimer

ContractWolf.io audits and reports should not be considered as a form of project's "advertisement" and does not cover any interaction and assessment from "project's contract" to "external contracts" such as Pancakeswap or similar.

ContractWolf does not provide any warranty on its released reports.

ContractWolf should not be used as a <u>decision</u> to invest into an audited project and is not affiliated nor partners to its audited contract projects.

ContractWolf provides transparent report to all its "clients" and to its "clients participants" and will not claim any guarantee of bug-free code within it's **SMART CONTRACT**.

ContractWolf presence is to analyze, audit and assess the client's smart contract's code.

Each company or projects should be liable to its security flaws and functionalities.

Network

BSC / Binance Smart Chain (BEP20 protocol)

Website

https://shiplay.io/

Twitter

https://twitter.com/shi_play

Telegram

t.me/shiplayio

Other channels

Discord: https://discord.com/invite/g8E4jP6W

Description

Shiba crypto world appear to be worlds apart when you think about it.

ShiPlay will be one of the most and Great games built as a play-to-earn game for anyone who has a deep love of SHIBA and MEME COINS and desires to earn tokens & NFT while enjoying his DOG.

ShiPlay is a blockchain-based. This is the beginning of a decentralized games ecosystem in which you can actively engage in a variety of ways. This universe combines four multibillion-dollar global markets: video games and NFT, all of which have been integrated using blockchain and contribute towards another multibillion-dollar business that has only recently begun to take off which is Metaverse

ContractWolf Engagement

14th of January 2022, **ShiPlay** engaged and agrees to audit their smart contract's code by ContractWolf. The goal of this engagement was to identify if there is a possibility of security flaws in the implementation of the contract or system.

ContractWolf will be focusing on contract issues and functionalities along with the projects claims from smart contract to their website, whitepaper and repository which has been provided by **ShiPlay.**

Contract link:
> https://bscscan.com/address/0x40671E899e915d39036935F7cFFe35a5348E0a5c

Logo

Contract Link

https://bscscan.com/address/0x40671E899e915d39036935F7cFFe35a53
48E0a5c

Risk level classification

Risk Level represents the classification or the probability that a certain function or threat that can exploit vulnerability and have an impact within the system or contract.
Risk Level is computed based on CVSS Version 3.0

Level	Value	Vulnerability
Critical	9 - 10	An exposure that can affect the contract functions in several events that can risk and disrupt the contract
High	7 - 8.9	An exposure that can affect the outcome when using the contract that can serve as an opening in manipulating the contract in an unwanted manner
Medium	4 - 6.9	An opening that could affect the outcome in executing the contract in a specific situation
Low	0.1 - 3.9	An opening but doesn't have an impact on the functionality of the contract
Informational	0	An opening that consists of information's but will not risk or affect the contract

Auditing Approach

Every line of code along with its functionalities will undergo manual review to check its security issues, quality, and contract scope of inheritance. The manual review will be done by our team that will document any issues that there were discovered.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:

- Review of the specifications, sources, and instructions provided to ContractWolf to make sure we understand the size, scope, and functionality of the smart contract.
- Manual review of code, our team will have a process of reading the code line-by-line with the intention of identifying potential vulnerabilities and security flaws.

2. Testing and automated analysis that includes:

- Testing the smart contract functions with common test cases and scenarios, to ensure that it returns the expected results.

3. Best practices review, the team will review the contract with the aim to improve efficiency, effectiveness, clarifications, maintainability, security, and control within the smart contract.

4. Recommendations to help the project take steps to secure the smart contract.

Used Code from other Frameworks/Smart Contracts (Direct Imports)

Imported Packages

- IERC20
- SafeMath
- Context
- Address
- SafeERC20
- Ownable
- IUniswapV2Factory
- IUniswapV2Route01
- IUniswapV2Route02
- Token

Description

Optimization enabled: Yes

Version: v0.8.6

Decimals: 18

Symbol: $ULAND

Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	1	3	4	2

Exposed Functions

Version	Public	Private
1.0	29	26

Version	External	Internal
1.0	37	21

State Variables

Version	Total	Public
1.0	40	24

Capabilities

| Version	Solidity Versions Observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	^0.8.4		Yes	Yes (3asm blocks)	No

Scope of Work

ShiPlay's team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract.

We will verify the following claims:

1. Correct implementation of Token standard.
2. Deployer cannot mint any new tokens.
3. Deployer cannot burn or lock user funds.
4. Deployer cannot pause the contract.
5. Overall checkup. (Smart Contract Security)

Inheritance Graph
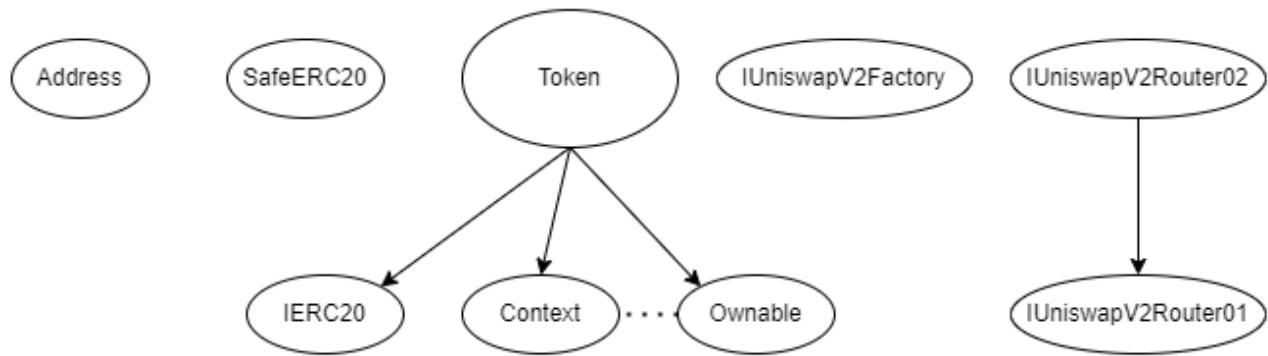
Verify Claims

Correct implementation of Token Standard

Tested	Verified
✓	✗

Function	Description	Exist	Tested	Verified
TotalSupply	Information about the total coin or token supply	✓	✓	✓
BalanceOf	Details on the account balance from a specified address	✓	✓	✓
Transfer	An action that transfers a specified amount of coin or token to a specified address	✓	✓	✓
TransferFrom	An action that transfers a specified amount of coin or token from a specified address	✓	✓	✓
Approve	Provides permission to withdraw specified number of coin or token from a specified address	✓	✓	✓
Allowance	Sets a specific number of coin or token that allows a specified address to utilize	✓	✓	✓

Optional implementation

Function	Description	Exist	Tested	Verified
renounceOwnership	Owner renounce ownership for more trust	✓	✓	✓

Deployer cannot mint any new tokens

Statement	Exist	Tested	Verified	File
Deployer cannot mint	—	—	—	Main

Max / Total supply: 1,000,000,000

Deployer cannot pause user funds

Statement	Exist	Tested	Verified
Deployer cannot pause	—	—	—

Deployer cannot burn user funds

Statement	Exist	Tested	Verified
Deployer cannot burn	—	—	—

Deployer cannot pause the contract

Statement	Exist	Tested	Verified
Deployer cannot pause	—	—	—

Overall Checkup (Smart Contract Security)

Tested	Verified
✓	✓

Legend

Attribute	Symbol
Verified / Checked	✓
Partly Verified	✗
Unverified / Not checked	⚑
Not Available	—

Write Functions of contract

1. approve

2. decreaseAllowance

3. deliver

4. excludeFromFee

5. excludeFromReward

6. includeInFee

7. includeInReward

8. increaseAllowance

9. lock

10. recoverBEP20

11. renounceOwnership

12. setAllFeePercent

13. setBuybackUpperLimit

14. setFeeWallet

15. setMaxTxPercent

16. setMaxWalletPercent

17. setSwapAndLiquifyEnabled

18. transfer

19. transferFrom

20. transferOwnership

21. unlock

SWC Attacks

ID	Title	Relationships	Status
SWC-136	Unencrypted Private Data On-Chain	CWE-767: Access to Critical Private Variable via Public Method	PASSED
SWC-135	Code With No Effects	CWE-1164: Irrelevant Code	NOT PASSED
SWC-134	Message call with hardcoded gas amount	CWE-655: Improper Initialization	PASSED
SWC-133	Hash Collisions with Multiple Variable Length Arguments	CWE-294: Authentication Bypass by Capture-replay	PASSED
SWC-132	Unexpected Ether balance	CWE-667: Improper Locking	PASSED
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	PASSED
SWC-130	Right-To Left Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	PASSED
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	PASSED
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	PASSED
SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	PASSED

SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	**PASSED**
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	**PASSED**
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	**PASSED**
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	**PASSED**
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	**PASSED**
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	**PASSED**
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	**PASSED**
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	**PASSED**
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	**PASSED**
SWC-116	Timestamp Dependence	CWE-829: Inclusion of	**PASSED**

		Functionality from Untrusted Control Sphere	
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	PASSED
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	PASSED
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	PASSED
SWC-112	Delegate call to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	PASSED
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	PASSED
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	PASSED
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED

SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	PASSED
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	PASSED
SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	PASSED
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	PASSED
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	PASSED
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	PASSED
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	PASSED
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED

AUDIT PASSED

Critical Issues

No critical issues found

High Issues

No high issues found

Medium Issues

No medium issues found

Low Issues

No low issues found

Informational Issues

No informational issues found

Function Issues

No informational issues found

Audit Comments

January 14, 2022

- Unnecessary comment containing state variable at line 726.
- No minting in contract.
- Read whole report for more information.