auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

XFLOKI

DATED : 27 July 23'

AUDIT SUMMARY

Project name –XFLOKI

Date: 27 July, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	1	0	0	0

USED TOOLS

Tools:

1- Manual Review:
A line by line code review has been performed by audit ace team.

2- BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/token/0x17f57FAc386b6923D744F5bC6ba074C7833a762E

Token Information

Token Name : X-Floki

Token Symbol: XFLOKI

Decimals: 18

Token Supply: 100,000,000

Token Address:
0x46aac59169084003Df6b2fF43Bb5984E34fc73d9

Checksum:
378d2f4a0a3311edde9ed10aa5f2fdb5e48fe6ef

Owner:
0xD28b0E54b0f2888aA7F25F0f45C177430043c2A0
(at time of writing the audit)

Deployer:
0xD28b0E54b0f2888aA7F25F0f45C177430043c2A0

TOKEN OVERVIEW

Fees:

Buy Fees: 0-10%

Sell Fees: 0-10%

Transfer Fees: 0-10%

Fees Privilege: owner

Ownership: owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: no

Blacklist: No

Other Privileges: Initial distribution of the tokens

modifying fees

enabling trades

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] Using throw

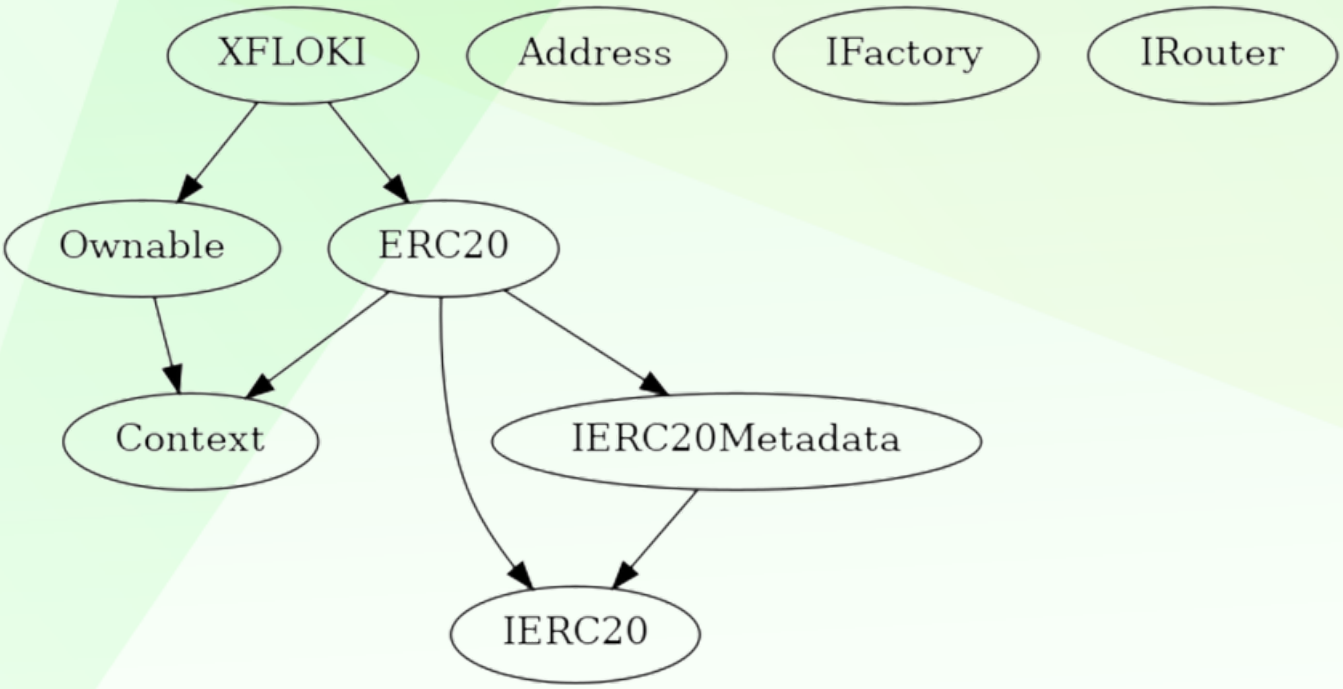
CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	1
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- **Owner is able to update fees within 0-10% for buy/sell/transfers**
- Owner is not able to blacklist an address
- Owner is not able to disable buy/sell/transfers
- Owner is not able to set max wallet limit and minimum wallet limits
- Owner is not able to mint new tokens
- **Owner must enable trades manually**

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20	Interface			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗

CONTRACT ASSESMENT

└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_tokengeneration	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
Address	Library			
└	sendValue	Internal 🔒	🛑	
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
IFactory	Interface			
└	createPair	External ❗	🛑	NO ❗
IRouter	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
XFLOKI	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗

CONTRACT ASSESMENT

└	transfer	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	Liquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForETH	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	updateLiquidityProvide	External ❗	🛑	onlyOwner
└	SetBuyTaxes	External ❗	🛑	onlyOwner
└	SetSellTaxes	External ❗	🛑	onlyOwner
└	UpdateMarketingWallet	External ❗	🛑	onlyOwner
└	updateLiquidityTreshhold	External ❗	🛑	onlyOwner
└	enableTrading	External ❗	🛑	onlyOwner
└	excludeFromFee	External ❗	🛑	onlyOwner
└	includeFromFee	External ❗	🛑	onlyOwner
└	rescueBNB	External ❗	🛑	NO ❗
└	rescueBEP2020	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO ❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

XFLOKI.excludeFromFee(address) (contracts/Token.sol#703-707) compares to a boolean constant:
        -require(bool,string)(exemptFee[_account] != true,Account is already excluded) (contracts/Token.sol#704)
XFLOKI.includeFromFee(address) (contracts/Token.sol#709-713) compares to a boolean constant:
        -require(bool,string)(exemptFee[_account] != false,Account is already included) (contracts/Token.sol#710)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Context._msgData() (contracts/Token.sol#23-26) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/Token.sol#16) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.21 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#333-341):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#336)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Variable ERC20._balances (contracts/Token.sol#72) is not in mixedCase
Variable ERC20._allowances (contracts/Token.sol#74) is not in mixedCase
Function IRouter.WETH() (contracts/Token.sol#391) is not in mixedCase
Event XFLOKIincludeFromFeeUpdated(address) (contracts/Token.sol#446) is not in CapWords
Function XFLOKI.Liquify(uint256,XFLOKI.Taxes) (contracts/Token.sol#585-623) is not in mixedCase
Function XFLOKI.SetBuyTaxes(uint256,uint256) (contracts/Token.sol#662-668) is not in mixedCase
Parameter XFLOKI.SetBuyTaxes(uint256,uint256)._marketing (contracts/Token.sol#663) is not in mixedCase
Parameter XFLOKI.SetBuyTaxes(uint256,uint256)._liquidity (contracts/Token.sol#664) is not in mixedCase
Function XFLOKI.SetSellTaxes(uint256,uint256) (contracts/Token.sol#670-676) is not in mixedCase
Parameter XFLOKI.SetSellTaxes(uint256,uint256)._marketing (contracts/Token.sol#671) is not in mixedCase
Parameter XFLOKI.SetSellTaxes(uint256,uint256)._liquidity (contracts/Token.sol#672) is not in mixedCase
Function XFLOKI.UpdateMarketingWallet(address) (contracts/Token.sol#678-683) is not in mixedCase
Parameter XFLOKI.UpdateMarketingWallet(address)._newWallet (contracts/Token.sol#678) is not in mixedCase
Parameter XFLOKI.updateLiquidityTreshhold(uint256).new_amount (contracts/Token.sol#685) is not in mixedCase
Parameter XFLOKI.excludeFromFee(address)._account (contracts/Token.sol#703) is not in mixedCase
Parameter XFLOKI.includeFromFee(address)._account (contracts/Token.sol#709) is not in mixedCase
Constant XFLOKI.Contract_Version (contracts/Token.sol#427) is not in UPPER_CASE_WITH_UNDERSCORES
Constant XFLOKI.Contract_Dev (contracts/Token.sol#428-429) is not in UPPER_CASE_WITH_UNDERSCORES
Constant XFLOKI.Contract_Edition (contracts/Token.sol#430) is not in UPPER_CASE_WITH_UNDERSCORES
Constant XFLOKI.deadWallet (contracts/Token.sol#433-434) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#24)" inContext (contracts/Token.sol#18-27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

XFLOKI.pair (contracts/Token.sol#418) should be immutable
XFLOKI.router (contracts/Token.sol#417) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0xd3e20006a55dc78fa1b925c569a6d1974b35267b96b030dd264f81406ab32fef

2- Buying when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0xb18ba7ef183ddf9e643a48e97ee5747c4b9c448ce0a0da43e2e39c3663a59788

3- Selling when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x9d455304deaa9abbc440b4112e47522c026fcff836caefdfdf5b55a986dfd7af

4- Transferring when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x79c0ada92729c6b667ab445d9dc7a64f1984d4329d21eec83d021f9442b2e67f

5- Buying when not excluded from fees (0-10% tax) (passed):

https://testnet.bscscan.com/tx/0xe2299aa1a531a7aa2c40cfa9fde8486352e11235aac2994c7f28247c007fafe5

6- Selling when not excluded from fees (0-10% tax) (passed):

https://testnet.bscscan.com/tx/0x51773e95c9f367fafc6cf5ac6c487042b55cf82d5a54f24ba1394f803d05a8e1

FUNCTIONAL TESTING

7- Transferring (0-10% tax) (passed):

https://testnet.bscscan.com/tx/0x504f5667c10871ecb0ffb48aa63a72550bb01dbe90139505f9fe60cca4937161

8- Internal swap(ETH sent to marketing wallet) (passed):

https://testnet.bscscan.com/tx/0x7c49c998f901bf8791010da7fa15258e07f52b9609e3b33accbe800e965ef5a6

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

function: enableTrading

Status: Resolved (Contract is owned by safu developer)

Overview:

Owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens.

```
function enableTrading() external onlyOwner {
  require(!tradingEnabled, "Cannot re-enable trading");
  tradingEnabled = true;
  providingLiquidity = true;
}
```

Suggestion

It's suggested to either enable trades prior to the presale, or transfer ownership of the contract to a certified pinsksale safu developer to guarantee enabling of trades.

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace