auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

XShiba

DATED : 3 august 23'

AUDIT SUMMARY

Project name – XShiba

Date: 3 august, 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/0x400FCC15eb7f19194e5C06C620b4ec852f1600Ba

Token Information

Token Name : XShiba

Token Symbol: XShiba

Decimals: 18

Token Supply: 100,000,000

Token Address:
0xC45EBB8aBDE7423E6B5F9d177E6a76d8623aAe6f

Checksum:
0d13ff50475c3fea38371e558f4b13bc5a383542

Owner:
0x57f7078cc69Ad1797DEE9EC750ac4b27A09aacb6
(at time of writing the audit)

Deployer:
0x57f7078cc69Ad1797DEE9EC750ac4b27A09aacb6

TOKEN OVERVIEW

Fees:

Buy Fees: 0%

Sell Fees: 0%

Transfer Fees: 0%

Fees Privilege: no fees

Ownership: owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: no

Blacklist: No

Other Privileges: Initial distribution of the tokens 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

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

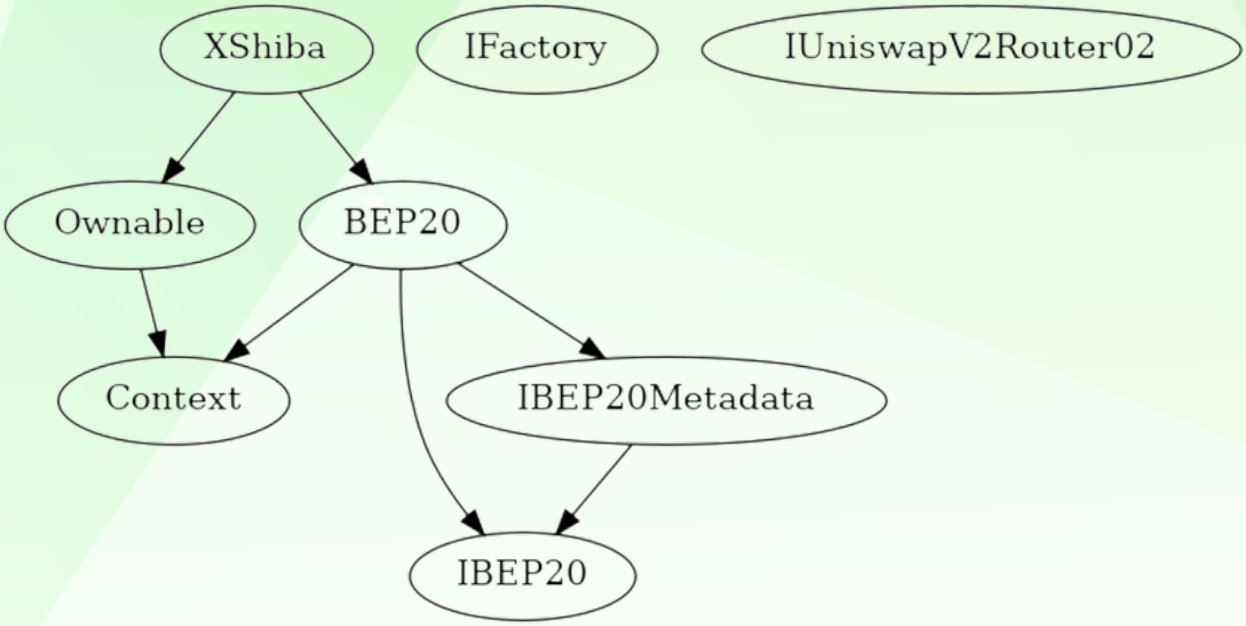
CLASSIFICATION OF RISK

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 not able to set fee on buy and transfers
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to mint new tokens
- Owner is not able to set max buy/sell/transfer
- Owner must enable trading for investors

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IBEP20	Interface			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
IBEP20Metadata	Interface	IBEP20		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
BEP20	Implementation	Context, IBEP20, IBEP20Metadata		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗

CONTRACT ASSESMENT

└	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 🔒	🛑	
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
IFactory	Interface			
└	createPair	External ❗	🛑	NO ❗
IUniswapV2Router02	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
XShiba	Implementation	BEP20, Ownable		
└	<Constructor>	Public ❗	🛑	BEP20
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	go_live	External ❗	🛑	onlyOwner
└	excludeFromFee	External ❗	🛑	onlyOwner
└	includeFromFee	External ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	rescueETH	External ❗	🛑	NO ❗
└	rescueERC20	External ❗	🛑	NO ❗
└	burnERC20	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💸	NO ❗

Legend

Symbol	Meaning
🛑	Function can modify state
💸	Function is payable

STATIC ANALYSIS

```
Reentrancy in XShiba.burnERC20(address,uint256) (contracts/Token.sol#418-422):
        External calls:
        - IBEP20(_tokenAdd).transfer(deadWallet,_amount) (contracts/Token.sol#420)
        Event emitted after the call(s):
        - ERC20TokensBurned() (contracts/Token.sol#421)
Reentrancy in XShiba.rescueERC20(address,uint256) (contracts/Token.sol#404-416):
        External calls:
        - IBEP20(_tokenAddy).transfer(owner(),_amount) (contracts/Token.sol#414)
        Event emitted after the call(s):
        - ERC20TokensRecovered(_amount) (contracts/Token.sol#415)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

XShiba.excludeFromFee(address) (contracts/Token.sol#385-389) compares to a boolean constant:
        -require(bool,string)(whitelist[_address] != true,Account is already excluded) (contracts/Token.sol#386)
XShiba.includeFromFee(address) (contracts/Token.sol#391-395) compares to a boolean constant:
        -require(bool,string)(whitelist[_address] != false,Account is already included) (contracts/Token.sol#392)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Context._msgData() (contracts/Token.sol#14-17) 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#7) 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

Variable BEP20._balances (contracts/Token.sol#54) is not in mixedCase
Variable BEP20._allowances (contracts/Token.sol#56) is not in mixedCase
Function IUniswapV2Router02.WETH() (contracts/Token.sol#238) is not in mixedCase
Event XShibaincludeFromFeeUpdated(address) (contracts/Token.sol#262) is not in CapWords
Function XShiba.go_live() (contracts/Token.sol#379-383) is not in mixedCase
Parameter XShiba.excludeFromFee(address)._address (contracts/Token.sol#385) is not in mixedCase
Parameter XShiba.includeFromFee(address)._address (contracts/Token.sol#391) is not in mixedCase
Parameter XShiba.rescueERC20(address,uint256)._tokenAddy (contracts/Token.sol#404) is not in mixedCase
Parameter XShiba.rescueERC20(address,uint256)._amount (contracts/Token.sol#404) is not in mixedCase
Parameter XShiba.burnERC20(address,uint256)._tokenAdd (contracts/Token.sol#418) is not in mixedCase
Parameter XShiba.burnERC20(address,uint256)._amount (contracts/Token.sol#418) is not in mixedCase
Constant XShiba.Contract_Version (contracts/Token.sol#250) is not in UPPER_CASE_WITH_UNDERSCORES
Constant XShiba.Contract_Dev_By (contracts/Token.sol#251) is not in UPPER_CASE_WITH_UNDERSCORES
Constant XShiba.Contract_Edition (contracts/Token.sol#252) is not in UPPER_CASE_WITH_UNDERSCORES
Constant XShiba.deadWallet (contracts/Token.sol#253-254) 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#15)" inContext (contracts/Token.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Reentrancy in XShiba.rescueETH() (contracts/Token.sol#397-402):
        External calls:
        - address(owner()).transfer(contractETHBalance) (contracts/Token.sol#400)
        Event emitted after the call(s):
        - ETHBalanceRecovered() (contracts/Token.sol#401)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

XShiba.uniswapV2Pair (contracts/Token.sol#243) should be immutable
XShiba.uniswapV2Router (contracts/Token.sol#242) 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/0x92575101220341966e0a838aa491f36712a3d5e450e78c71a8c3c3dec6ec0edb

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

https://testnet.bscscan.com/tx/0x940292722cbd3d2004df4d52345ec8d8087a19ac61836d52f0146db32bc0df4f

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

https://testnet.bscscan.com/tx/0xc985485871612c9c0ae58528157e8c1311039266d5a785222524aaf93d68f980

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

https://testnet.bscscan.com/tx/0xd9bb56f0aee2ce8d76a53169d32715eb96e4001606ad779b764bc42a9a98e54b

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

https://testnet.bscscan.com/tx/0xf019087f3b65e64ce1e5392637e1cc0809d67ae4a1bf080d0722572ce781e4df

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

https://testnet.bscscan.com/tx/0x76b48443f8eeca92976aec4fec342c121e9a29309edef4263b7224ac9135a0a7

FUNCTIONAL TESTING

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

https://testnet.bscscan.com/tx/0xb401670911e1c72f7a3a0545f38230666fc040ca3481b0b8e5cef8454c937114

High Risk

Centralization – Enabling Trades

Severity: High

function: go_live

Status: Resolved (Contract By SAFU Dev)

Overview:

Owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens (even owner of whitelisted wallets)

```
function go_live() external onlyOwner {
  require(!tradingEnabled, "Trading is already enabled");
  tradingEnabled = true;
  emit TradingOpenUpdated();
}
```

Suggestion

Its suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guearanee 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