auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

BoysClub

DATED : 25 September 23'

AUDIT SUMMARY

Project name – BoysClub

Date: 25 September 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	0	0	0	0

USED TOOLS

Tools:

1.Code Comparison:

We used specialized tools to perform a line-by-line comparison between the project's code and that of Uniswap V2 to identify any differences.

2.Differential Analysis:

Our audit team conducted a thorough review of the differentials to assess whether they introduce any security vulnerabilities or logical errors.

3.Additional Modules:

Any additional smart contracts, not part of the original Uniswap V2, were audited as separate entities, following our standard auditing procedures.

4. Testnet version:

https://testnet.bscscan.com/token/0x915c4E8A7cc98341C52246e02D2d5cd9F40A5A3F

Token Information

Token Address :
0x0bEBfF25C6a3de96a84D56B1796bA4079b65CAF4

Name: BoysClub

Symbol: BOYSBOT

Decimals: 18

Network: Ethereum mainnet

Token Type: ERC20

Owner: 0x0Bbd2B707F134F81070875425dAB0207D9503424

Deployer:
0x0Bbd2B707F134F81070875425dAB0207D9503424

Token Supply: 10,000,000
Checksum:
6394dabf245084fa42b09f99a661d9127008337b

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/0x915c4E8A7cc98341C52246e02D2d5cd9F40A5A3F

TOKEN OVERVIEW

buy fee: 0%

Sell fee: 0%

transfer fee: 0%

Fee Privilege: no fees

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

Other Privileges:

- Initial distribution of the tokens

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

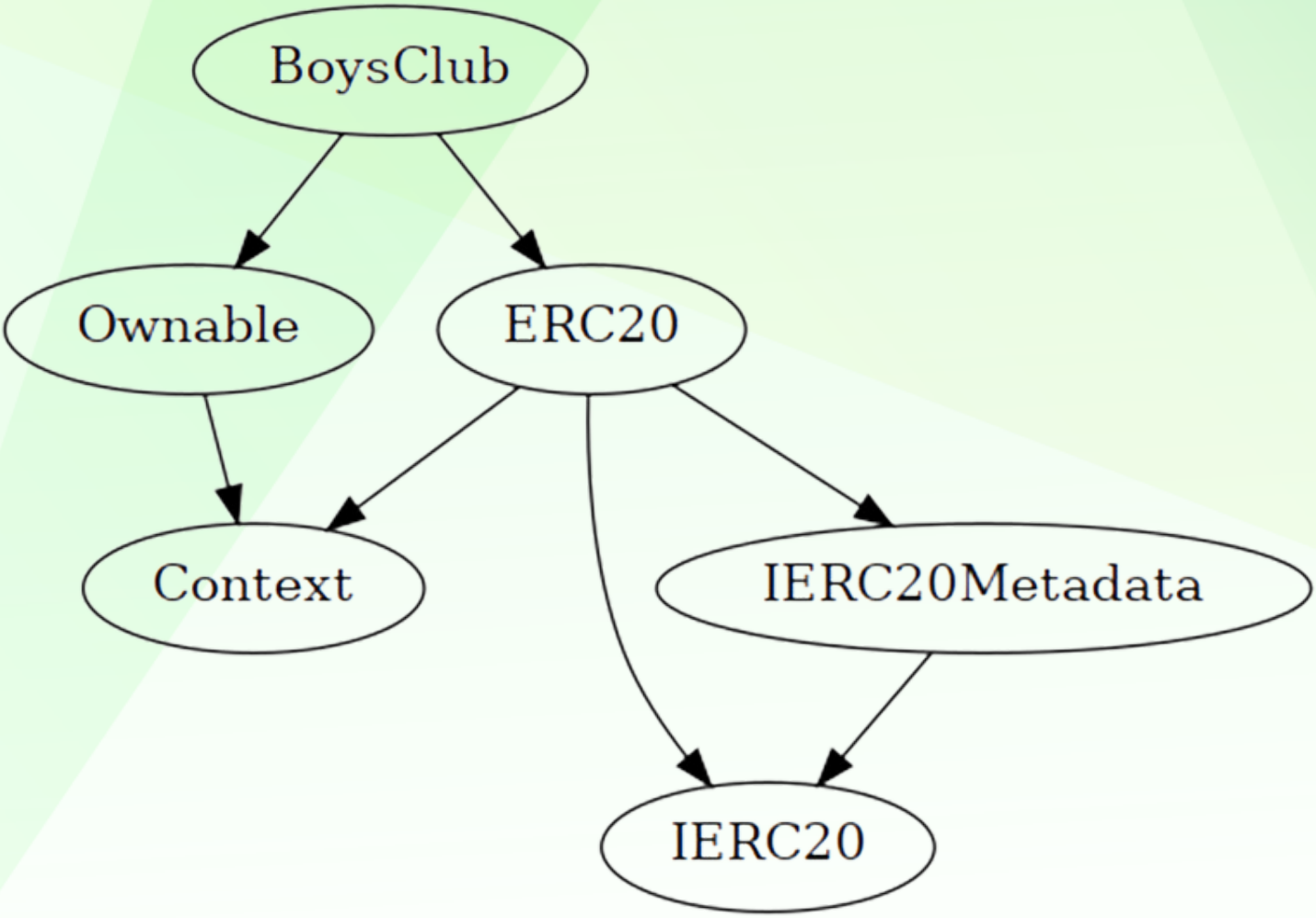
CLASSIFICATION OF RISK

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	0
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set buy/sell/transfer fees
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner is not able to mint new tokens

STATIC ANALYSIS

```
INFO:Detectors:
Different versions of Solidity are used:
        - Version used: ['^0.8.0', '^0.8.17']
        - ^0.8.0 (contracts/Token.sol#11)
        - ^0.8.0 (contracts/Token.sol#38)
        - ^0.8.0 (contracts/Token.sol#123)
        - ^0.8.0 (contracts/Token.sol#204)
        - ^0.8.0 (contracts/Token.sol#234)
        - ^0.8.17 (contracts/Token.sol#599)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
INFO:Detectors:
Context._msgData() (contracts/Token.sol#28-30) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#506-522) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (contracts/Token.sol#11) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#38) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#123) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#204) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#234) allows old versions
Pragma version^0.8.17 (contracts/Token.sol#599) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
BoysClub.constructor() (contracts/Token.sol#604-606) uses literals with too many digits:
        - _mint(msg.sender,10000000 * 10 ** decimals()) (contracts/Token.sol#605)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

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

No major issues were found in the output

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	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 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
BoysClub	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

FUNCTIONAL TESTING

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0x4acf4a9919c0e95099a91b1546c20c51a3efc062ed0db529ab613bbe4133c446

2- Buying (0% tax) (passed):

https://testnet.bscscan.com/tx/0x247306be181528e59297d3400d4e65eac931724fcf975af9444f1f7973f9ce75

3- Selling (0% tax) (passed):

https://testnet.bscscan.com/tx/0x525d45d954be3226b706cbc10fa00142563e9935f5b7375753f787f28d5035e3

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

https://testnet.bscscan.com/tx/0xf91be04e5194de258629cebf0acefed54b9b4d293ee74b3bd321b11b03172026

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