auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

LAUGH COIN

DATED : 19 Jan 2023

AUDIT SUMMARY

Project name – LAUGH COIN

Date: 19 Jan 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	2	3
Acknowledged	0	0	0	0	0
Resolved	0	0	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/address/0x2d5761827310b52579c732bd496face7b9b37844#code

Token Information

Token Address:
0x0213E492E9936e4DcFa0F2aa6c4BaD53bCAf8BA9

Name: LAUGH COIN

Symbol: LAFF

Decimals: 18

Network: BscScan

Token Type: BEP-20

Owner:
0x1d4deae1f2b8353e9a6ccf0c3c16bb0598c1fc91

Deployer:
0xe0d5cc36192430a8f4c3d42e56b9968b6fa83d56

Token Supply: 1000,000,000,000

Checksum: fc6659e84744e0102ab19c1d1e78a822

Testnet:
https://testnet.bscscan.com/address/0x2d5761827310b52579c732bd496face7b9b37844#code

AUDIT METHODOLOGY

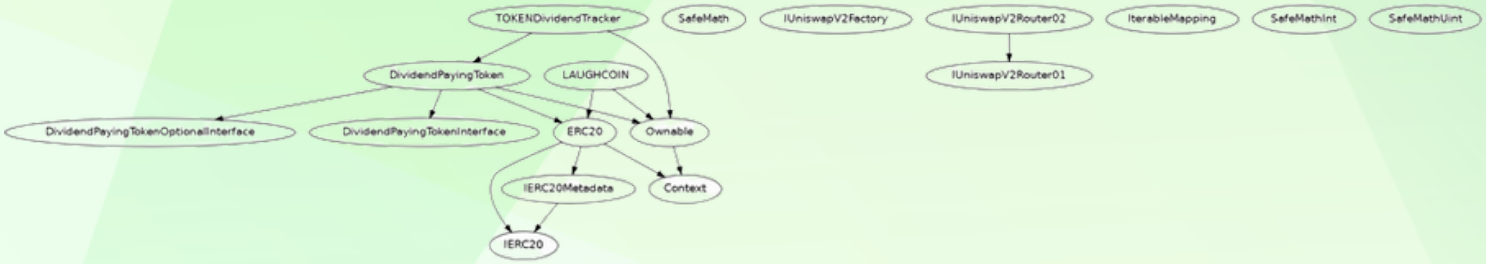
The auditing process will follow a routine as special considerations by Auditace:

- 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

INHERITANCE TREE



POINTS TO NOTE

- The owner can transfer ownership.
- The owner can renounce ownership.
-The owner can distribute reward tokens.
- The owner can set the Buy/Sell fees more than 25%.
-The owner can update reward token.
- The owner can withdraw BNB.
-The owner can update uniswapV2router.
- The owner can exclude wallets from fees.
-The owner can set wallet addresses.
- The owner can set automated market maker pair address.
-The owner can set swap tokens amount.
- The owner can set max buy amount.
-The owner can update gas for processing fees.
- The owner can exclude address from dividends.
-The owner can set balance.

STATIC ANALYSIS

```
INFO:Detectors:
DividendPayingToken.constructor(string,string,address)._name (LAUGHCOIN.sol#1326) shadows:
        - ERC20._name (LAUGHCOIN.sol#188) (state variable)
DividendPayingToken.constructor(string,string,address)._symbol (LAUGHCOIN.sol#1327) shadows:
        - ERC20._symbol (LAUGHCOIN.sol#189) (state variable)
DividendPayingToken.dividendOf(address)._owner (LAUGHCOIN.sol#1378) shadows:
        - Ownable._owner (LAUGHCOIN.sol#589) (state variable)
DividendPayingToken.withdrawableDividendOf(address)._owner (LAUGHCOIN.sol#1382) shadows:
        - Ownable._owner (LAUGHCOIN.sol#589) (state variable)
DividendPayingToken.withdrawnDividendOf(address)._owner (LAUGHCOIN.sol#1391) shadows:
        - Ownable._owner (LAUGHCOIN.sol#589) (state variable)
DividendPayingToken.accumulativeDividendOf(address)._owner (LAUGHCOIN.sol#1400) shadows:
        - Ownable._owner (LAUGHCOIN.sol#589) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
LAUGHCOIN.setBuyFees(uint16,uint16) (LAUGHCOIN.sol#1675-1693) should emit an event for:
        - totalBuyFee = buyFee.reward + buyFee.marketing (LAUGHCOIN.sol#1686-1689)
LAUGHCOIN.setSellFees(uint16,uint16) (LAUGHCOIN.sol#1695-1712) should emit an event for:
        - totalSellFee = sellFee.reward + sellFee.marketing (LAUGHCOIN.sol#1706-1709)
LAUGHCOIN.setSwapTokens(uint256) (LAUGHCOIN.sol#1721-1723) should emit an event for:
        - swapTokensAtAmount = amount * 10 ** 18 (LAUGHCOIN.sol#1722)
LAUGHCOIN.setMaxBuy(uint256) (LAUGHCOIN.sol#1725-1731) should emit an event for:
        - maxBuyAmount = amount * 10 ** 18 (LAUGHCOIN.sol#1730)
LAUGHCOIN.setMaxSell(uint256) (LAUGHCOIN.sol#1739-1745) should emit an event for:
        - maxSellAmount = amount * 10 ** 18 (LAUGHCOIN.sol#1744)
LAUGHCOIN.setMaxWallet(uint256) (LAUGHCOIN.sol#1747-1753) should emit an event for:
        - maxWalletAmount = amount * 10 ** 18 (LAUGHCOIN.sol#1752)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
LAUGHCOIN.updateRewardToken(address).newToken (LAUGHCOIN.sol#1615) lacks a zero-check on :
                - RewardToken = newToken (LAUGHCOIN.sol#1625)
LAUGHCOIN.updateUniswapV2Router(address)._uniswapV2Pair (LAUGHCOIN.sol#1648-1649) lacks a zero-check on :
                - uniswapV2Pair = _uniswapV2Pair (LAUGHCOIN.sol#1650)
LAUGHCOIN.setWallet(address).mkt (LAUGHCOIN.sol#1670) lacks a zero-check on :
                - marketingWallet = mkt (LAUGHCOIN.sol#1671)
DividendPayingToken.constructor(string,string,address)._rewardToken (LAUGHCOIN.sol#1328) lacks a zero-check on :
                - RewardToken = _rewardToken (LAUGHCOIN.sol#1330)
```

```
INFO:Detectors:
LAUGHCOIN.swapAndSendToFee(uint256,uint16) (LAUGHCOIN.sol#1924-1943) tries to limit the gas of an external call that controls implicit decoding
        (status) = marketingWallet.call{gas: 30000,value: marketingShare}() (LAUGHCOIN.sol#1936-1939)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#return-bomb
INFO:Detectors:
TOKENDividendTracker.getAccount(address) (LAUGHCOIN.sol#2092-2139) uses timestamp for comparisons
        Dangerous comparisons:
        - nextClaimTime > block.timestamp (LAUGHCOIN.sol#2136-2138)
TOKENDividendTracker.canAutoClaim(uint256) (LAUGHCOIN.sol#2173-2179) uses timestamp for comparisons
        Dangerous comparisons:
        - lastClaimTime > block.timestamp (LAUGHCOIN.sol#2174)
        - block.timestamp.sub(lastClaimTime) >= claimWait (LAUGHCOIN.sol#2178)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Different versions of Solidity are used:
        - Version used: ['>=0.5.0', '>=0.6.2', '^0.8.19']
        - >=0.5.0 (LAUGHCOIN.sol#903)
        - >=0.6.2 (LAUGHCOIN.sol#937)
        - >=0.6.2 (LAUGHCOIN.sol#1100)
        - ^0.8.19 (LAUGHCOIN.sol#9)
        - ^0.8.19 (LAUGHCOIN.sol#100)
        - ^0.8.19 (LAUGHCOIN.sol#128)
        - ^0.8.19 (LAUGHCOIN.sol#154)
        - ^0.8.19 (LAUGHCOIN.sol#574)
        - ^0.8.19 (LAUGHCOIN.sol#656)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
INFO:Detectors:
LAUGHCOIN._transfer(address,address,uint256) (LAUGHCOIN.sol#1817-1922) has a high cyclomatic complexity (15).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
```

STATIC ANALYSIS

```
INFO:Detectors:
Low level call in LAUGHCOIN.swapAndSendToFee(uint256,uint16) (LAUGHCOIN.sol#1924-1943):
        - (status) = marketingWallet.call{gas: 30000,value: marketingShare}() (LAUGHCOIN.sol#1936-1939)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Router01.WETH() (LAUGHCOIN.sol#942) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (LAUGHCOIN.sol#1378) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (LAUGHCOIN.sol#1382) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (LAUGHCOIN.sol#1391) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (LAUGHCOIN.sol#1400) is not in mixedCase
Variable DividendPayingToken.RewardToken (LAUGHCOIN.sol#1314) is not in mixedCase
Constant DividendPayingToken.magnitude (LAUGHCOIN.sol#1316) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter LAUGHCOIN.withdrawForeignTokens(address,uint256)._token (LAUGHCOIN.sol#1631) is not in mixedCase
Parameter LAUGHCOIN.withdrawForeignTokens(address,uint256)._amount (LAUGHCOIN.sol#1631) is not in mixedCase
Parameter LAUGHCOIN.withdrawBNB(uint256)._amount (LAUGHCOIN.sol#1641) is not in mixedCase
Constant LAUGHCOIN.deadWallet (LAUGHCOIN.sol#1497) is not in UPPER_CASE_WITH_UNDERSCORES
Variable LAUGHCOIN.RewardToken (LAUGHCOIN.sol#1499-1500) is not in mixedCase
Parameter TOKENDividendTracker.getAccount(address)._account (LAUGHCOIN.sol#2092) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (LAUGHCOIN.sol#947) is too similar
to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (LAUGHCOIN.sol#948)
Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (LAUGHCOIN.sol#1354) is too similar to TOKENDividendTracker.getAccount(a
ddress).withdrawableDividends (LAUGHCOIN.sol#2099)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
LAUGHCOIN.updateGasForProcessing(uint256) (LAUGHCOIN.sol#1765-1770) uses literals with too many digits:
        - require(bool)(newValue >= 200000 && newValue <= 500000) (LAUGHCOIN.sol#1766)
LAUGHCOIN.slitherConstructorVariables() (LAUGHCOIN.sol#1463-2015) uses literals with too many digits:
        - gasForProcessing = 200000 (LAUGHCOIN.sol#1512)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
SafeMathInt.MAX_INT256 (LAUGHCOIN.sol#1255) is never used in SafeMathInt (LAUGHCOIN.sol#1253-1292)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Slither:LAUGHCOIN.sol analyzed (17 contracts with 93 detectors), 86 result(s) found
```

FUNCTIONAL TESTING

1- Approve (passed):

https://testnet.bscscan.com/tx/0x775ec3d811062f7d746c8e29eb0f808aacda576fe8a209af2a02470eef7a8ab4

2- Increase Allowance (passed):

https://testnet.bscscan.com/tx/0x2169dbd51caca6afbe667f73c45ef3632dd21800d7a253920d8d35b7b167c0ee

3- Decrease Allowance (passed):

https://testnet.bscscan.com/tx/0x4f9e913d6bd6cf873e35ffdd735a77847a64cd7656e9731bfbd0a21cdf879c7a

4- Exclude From Dividends (passed):

https://testnet.bscscan.com/tx/0x0be67aa185eb13047e9fd109ddd4deec203269a302aa6a74b078bfe31ef1f5fb

5- Exclude From Fees (passed):

https://testnet.bscscan.com/tx/0xce5a3892ba08c3fadac7eb553534cbc2e72292a6d02bc73459b4c34c58d86dd8

6- Set Sell Fees (passed):

https://testnet.bscscan.com/tx/0x0ebd723012fe7cb951d14eca3ad313518d3d21b4bf9d1a4a6d8ec3ab9af33744

7- Set Buy Fees (passed):

https://testnet.bscscan.com/tx/0x9cb0eee3b90e763621f8b8e9bfe0e32d4b4b172eff101b13822e8ae562d3832b

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	2
Gas Optimization / Suggestions	3

MANUAL TESTING

Centralization - Missing Events
Severity: Low
Subject: Missing Events
Status: Open

Overview:
They serve as a mechanism for emitting and recording data onto the blockchain, makingit transparent and easily accessible.

```
function updateUniswapV2Router(address
newAddress) public onlyOwner { emit
UpdateUniswapV2Router(newAddress,
address(uniswapV2Router)); uniswapV2Router =
IUniswapV2Router02(newAddress);
 address _uniswapV2Pair =
IUniswapV2Factory(uniswapV2Router.factory())
 .createPair(address(this), uniswapV2Router.WETH());
uniswapV2Pair = _uniswapV2Pair;
}
function setWallet(address mkt) external onlyOwner {
marketingWallet = mkt;

}
function setBuyFees(
```

MANUAL TESTING

```
uint16 reward, uint16 market

) external onlyOwner {

buyFee.reward = reward; buyFee.marketing = market;

totalBuyFee =

buyFee.reward + buyFee.marketing ;
require(totalBuyFee <= 300, "30% limit");
}
function setSellFees(

uint16 reward, uint16 market
) external onlyOwner {

sellFee.reward = reward; sellFee.marketing = market;
totalSelFee =
sellFee.reward + sellFee.marketing ;

require(totalSellFee <= 300, "30% limit");
}
function setSwapTokens(uint256 amount) external onlyOwner {
swapTokensAtAmount = amount * 10**18;
}
function setMaxBuy(uint256 amount) external onlyOwner { maxBuyAmount =
amount * 10**18;
}
function openTrading() external onlyOwner { enableTrading = true;
}
function setMaxSell(uint256 amount) external onlyOwner { maxSellAmount =
amount * 10**18;
}
function setMaxWallet(uint256 amount) external onlyOwner { maxWalletAmount
= amount * 10**18;
}
function updateClaimWait(uint256 claimWait) external onlyOwner {
dividendTracker.updateClaimWait(claimWait);
}
```

MANUAL TESTING

Centralization - Missing Zero Address
Severity: Low
Subject: Zero Check
Status: Open

Overview:
functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.

```
function updateRewardToken(address newToken)
public onlyOwner { TOKENDividendTracker
newDividendTracker = new TOKENDividendTracker(
newToken
);


newDividendTracker.excludeFromDividends(address(n
ewDividendTracker));
newDividendTracker.excludeFromDividends(address(t
his));
newDividendTracker.excludeFromDividends(owner());
newDividendTracker.excludeFromDividends(address(u
niswapV2Router));
```

MANUAL TESTING

```
RewardToken = newToken; dividendTracker =
newDividendTracker;

emit UpdateDividendTracker(newToken,
address(dividendTracker));
}
function updateUniswapV2Router(address
newAddress) public onlyOwner { emit
UpdateUniswapV2Router(newAddress,
address(uniswapV2Router)); uniswapV2Router
= IUniswapV2Router02(newAddress);
 address _uniswapV2Pair =
IUniswapV2Factory(uniswapV2Router.factory()
)
 .createPair(address(this),
uniswapV2Router.WETH()); uniswapV2Pair =
_uniswapV2Pair;
}
function setWallet(address mkt) external
onlyOwner { marketingWallet = mkt;
```

MANUAL TESTING

Optimization

Severity: Informational

Subject: Floating Pragma.

Status: Open

Overview:

It is considered best practice to pick one compiler versionand stick with it. With a floatingpragma, contracts may accidentally be deployed using an outdated.

pragma solidity pragma solidity^0.8.19;

Suggestion:

Adding the latest constant versionof solidity is recommended, as this prevents the unintentional deployment of a contract with an outdated compiler that contains unresolved bugs.

MANUAL TESTING

Optimization

Severity: Informational

Subject: Remove Safe Math

Status: Open

Line: 669-900

Overview:

compiler versionabove 0.8.0 can control arithmetic overflow/underflow, It is recommended to remove the unwanted code to avoid high gas fees.

MANUAL TESTING

Optimization

Severity: Informational

Subject: Remove unused code.

Status: Open

Overview:

Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them.

```
event LiquidityWalletUpdated(
address indexed newLiquidityWallet,
address indexed oldLiquidityWallet
);
event SwapAndLiquify(
uint256 tokensSwapped,
uint256 ethReceived,
uint256 tokensIntoLiqudity
);
```

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