AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

$QUACK

DATED : 29 June 23'

FUNCTIONAL TESTING

Centralization – Trades must be enabled

Severity: **High**

function: OpenTrading

Status: Not Resolved

Overview:

The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

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

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.

3. Transfer ownership to a trusted and valid 3rd party in order to guarantee enabling of the trades

FUNCTIONAL TESTING

Centralization – Excessive fees

Severity: **High**

function: BuyTaxes - SellTaxes

Status: Not Resolved

Overview:

Owner is able to set 100% tax on buy/sell/transfers.

```
function BuyTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    taxes = Taxes(_marketing, _liquidity);
}

function SellTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    sellTaxes = Taxes(_marketing, _liquidity);
}
```

Suggestion

to mitigate this issue ensure that fees are within a reasonable and safe range. 0-10% is suggested by pinksale safu criteria

```
function BuyTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    require(_marketing + _liquidity <= 10, "Cant set fees over 10%");
    taxes = Taxes(_marketing, _liquidity);
}

function SellTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    require(_marketing + _liquidity <= 10, "Cant set fees over 10%");
    sellTaxes = Taxes(_marketing, _liquidity);
}
```

AUDIT SUMMARY

Project name – $QUACK

Date: 29 June, 2023

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

Audit Status: **Passed with High Risk**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	2	0	0	0
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/token/0x6BbC026E1E5bDe 0B94821Fd7d1E0b85e19d56628

Token Information

Token Name : QUACK TV

Token Symbol: $QUACK

Decimals: 18

Token Supply: 1,000,000,000

Token Address:
0x04D749487Ad4707509eC4f30d40FC98f06259cFa

Checksum:
87336c8f273d3e3842b2e7ef827ded41952ed023

Owner:
0x32B280661E895251Dd522C08935e8fbfaAA8b028
(at time of writing the audit)

Deployer:
0x32B280661E895251Dd522C08935e8fbfaAA8b028

TOKEN OVERVIEW

Fees:

Buy Fees: 0-100%

Sell Fees: 0-100%

Transfer Fees: 0-100%

Fees Privilege: Owner

Ownership: Owned

Minting: none

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: - initial distribution of tokens

- including or excluding from fees

- changing swap threshold

- changing 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

✅ Return values of low-level calls

✅ **Gasless Send**

✅ Private modifier

✅ Using block.timestamp

✅ Multiple Sends

✅ Re-entrancy

✅ Using Suicide

✅ Tautology or contradiction

✅ Gas Limitand Loops

✅ Timestamp Dependence

✅ Address hardcoded

✅ Revert/require functions

✅ Exception Disorder

✅ Use of tx.origin

✅ Using inline assembly

✅ Integer overflow/underflow

✅ Divide before multiply

✅ Dangerous strict equalities

✅ Missing Zero Address Validation

✅ Using SHA3

✅ Compiler version not fixed

✅ 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	2
◆ Medium-Risk	0
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	0

INHERITANCE TREE
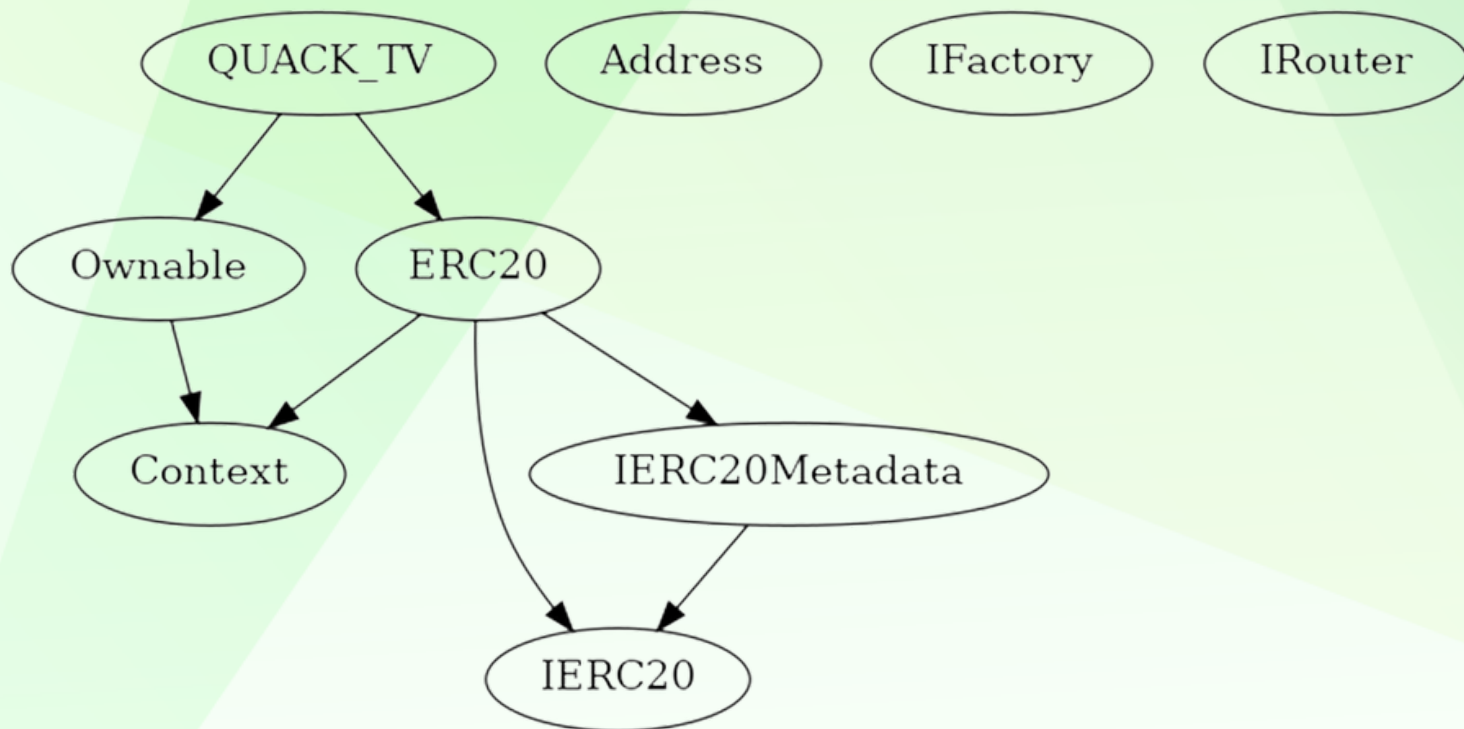
POINTS TO NOTE

- **Owner is able to set buy/transfer tax up to 100%**
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to set max wallet/buy/sell/transfer
- 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 ❗
└	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 ❗

CONTRACT ASSESMENT

└	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❗
QUACK_TV	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🔴	ERC20
└	approve	Public ❗	🔴	NO❗
└	transferFrom	Public ❗	🔴	NO❗
└	increaseAllowance	Public ❗	🔴	NO❗
└	decreaseAllowance	Public ❗	🔴	NO❗
└	transfer	Public ❗	🔴	NO❗
└	_transfer	Internal 🔐	🔴	
└	Liquify	Private 🔐	🔴	lockTheSwap
└	swapTokensForETH	Private 🔐	🔴	
└	addLiquidity	Private 🔐	🔴	
└	updateLiquidityProvide	External ❗	🔴	onlyOwner
└	updateLiquidityTreshhold	External ❗	🔴	onlyOwner
└	enableTrading	External ❗	🔴	onlyOwner
└	BuyTaxes	External ❗	🔴	onlyOwner
└	SellTaxes	External ❗	🔴	onlyOwner
└	updateExemptFee	External ❗	🔴	onlyOwner
└	UpdateMarketingWallet	External ❗	🔴	onlyOwner
└	ClearETHBalance	External ❗	🔴	NO❗
└	ClearERC20Tokens	External ❗	🔴	onlyOwner
└	<Receive Ether>	External ❗	💲	NO❗

Legend

Symbol	Meaning
🔴	Function can modify state
💲	Function is payable

STATIC ANALYSIS

```
Context._msgData() (contracts/Token.sol#17-20) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

QUACK_TV.BuyFee (contracts/Token.sol#414) is set pre-construction with a non-constant function or state variable:
        - taxes.marketing + taxes.liquidity
QUACK_TV.SellFee (contracts/Token.sol#415) is set pre-construction with a non-constant function or state variable:
        - sellTaxes.marketing + sellTaxes.liquidity
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Pragma version^0.8.17 (contracts/Token.sol#10) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.20 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#317-322):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#320)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Variable ERC20._balances (contracts/Token.sol#63) is not in mixedCase
Variable ERC20._allowances (contracts/Token.sol#65) is not in mixedCase
Function IRouter.WETH() (contracts/Token.sol#366) is not in mixedCase
Contract QUACK_TV (contracts/Token.sol#393-641) is not in CapWords
Function QUACK_TV.Liquify(uint256,QUACK_TV.Taxes) (contracts/Token.sol#530-561) is not in mixedCase
Parameter QUACK_TV.updateLiquidityTreshhold(uint256).new_amount (contracts/Token.sol#593) is not in mixedCase
Function QUACK_TV.BuyTaxes(uint256,uint256) (contracts/Token.sol#605-607) is not in mixedCase
Parameter QUACK_TV.BuyTaxes(uint256,uint256)._marketing (contracts/Token.sol#605) is not in mixedCase
Parameter QUACK_TV.BuyTaxes(uint256,uint256)._liquidity (contracts/Token.sol#605) is not in mixedCase
Function QUACK_TV.SellTaxes(uint256,uint256) (contracts/Token.sol#609-611) is not in mixedCase
Parameter QUACK_TV.SellTaxes(uint256,uint256)._marketing (contracts/Token.sol#609) is not in mixedCase
Parameter QUACK_TV.SellTaxes(uint256,uint256)._liquidity (contracts/Token.sol#609) is not in mixedCase
Parameter QUACK_TV.updateExemptFee(address,bool)._address (contracts/Token.sol#613) is not in mixedCase
Function QUACK_TV.UpdateMarketingWallet(address) (contracts/Token.sol#617-622) is not in mixedCase
Parameter QUACK_TV.UpdateMarketingWallet(address)._newWallet (contracts/Token.sol#617) is not in mixedCase
Function QUACK_TV.ClearETHBalance() (contracts/Token.sol#624-629) is not in mixedCase
Function QUACK_TV.ClearERC20Tokens(address,uint256) (contracts/Token.sol#631-636) is not in mixedCase
Parameter QUACK_TV.ClearERC20Tokens(address,uint256)._tokenAddy (contracts/Token.sol#631) is not in mixedCase
Parameter QUACK_TV.ClearERC20Tokens(address,uint256)._amount (contracts/Token.sol#631) is not in mixedCase
Constant QUACK_TV.deadWallet (contracts/Token.sol#404) is not in UPPER_CASE_WITH_UNDERSCORES
Variable QUACK_TV.BuyFee (contracts/Token.sol#414) is not in mixedCase
Variable QUACK_TV.SellFee (contracts/Token.sol#415) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#18)" inContext (contracts/Token.sol#12-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

QUACK_TV.constructor() (contracts/Token.sol#427-439) uses literals with too many digits:
        - _tokengeneration(msg.sender,1000000000 * 10 ** decimals()) (contracts/Token.sol#428)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

QUACK_TV.BuyFee (contracts/Token.sol#414) should be immutable
QUACK_TV.SellFee (contracts/Token.sol#415) should be immutable
QUACK_TV.pair (contracts/Token.sol#396) should be immutable
QUACK_TV.router (contracts/Token.sol#395) 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

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0xa946cbe13d3231dcc688a5237fd15e8f4de8c3ff0efa7d1a6ee5124318e30e4b

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

https://testnet.bscscan.com/tx/0x429360da65f298910bdb861650c280e25df4ad2598adb0e03ea88d38b7fd21db

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

https://testnet.bscscan.com/tx/0x94276f31241f1ce17c7d807de640d9abec7c4d87d0dcb85b20d452fe6aff3824

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

https://testnet.bscscan.com/tx/0x18536f0088abc4f2a2e4c6a3b8bfa3bc0c5821a35bcc59a167273f3e0d498efd

5- Buying when not excluded from fees (0-100% tax) (passed):

https://testnet.bscscan.com/tx/0x2353f8be72c3d2e961ead59bf5999ae94349fd9b772a21e6ba0ede88276721ca

6- Selling when not excluded from fees (0-100% tax) (passed):

https://testnet.bscscan.com/tx/0x954f6968e129ab16b36b776b4a4bf6257f6d63c1a8c7225deeabb3fcbc7c2f7c

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0-100% tax) (passed):

https://testnet.bscscan.com/tx/0x9db9bb01c21fadcc2a7c6e14e90f50c9995f7f22f0cabd46ab199de68e2e0d5a

8- Internal swap (auto-liquidty, marketing wallet received ETH)
(passed):

https://testnet.bscscan.com/tx/0x954f6968e129ab16b36b776b4a4bf6257f6d63c1a8c7225deeabb3fcbc7c2f7c

FUNCTIONAL TESTING

Centralization – Trades must be enabled

Severity: **High**
function: OpenTrading
Status: Not Resolved

Overview:

The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

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

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.

3. Transfer ownership to a trusted and valid 3rd party in order to guarantee enabling of the trades

FUNCTIONAL TESTING

Centralization – Excessive fees

Severity: **High**

function: BuyTaxes - SellTaxes

Status: Not Resolved

Overview:

Owner is able to set 100% tax on buy/sell/transfers.

```
function BuyTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    taxes = Taxes(_marketing, _liquidity);
}

function SellTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    sellTaxes = Taxes(_marketing, _liquidity);
}
```

Suggestion

to mitigate this issue ensure that fees are within a reasonable and safe range. 0-10% is suggested by pinksale safu criteria

```
function BuyTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    require(_marketing + _liquidity <= 10, "Cant set fees over 10%");
    taxes = Taxes(_marketing, _liquidity);
}

function SellTaxes( uint256 _marketing, uint256 _liquidity) external onlyOwner {
    require(_marketing + _liquidity <= 10, "Cant set fees over 10%");
    sellTaxes = Taxes(_marketing, _liquidity);
}
```

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