



Smart Contract Audit

FOR
ETHK

DATED : 4 august 23'



MANUAL TESTING

Centralization – Enabling Trades

Severity: **High**

function: enableTrading

Status: Open

Overview:

The 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 enableTrading() public onlyOwner {  
    require(!tradingEnabled, "Trading is already enabled");  
    tradingEnabled = true;  
}
```

Suggestion

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



AUDIT SUMMARY

Project name -ETHKing

Date: 4 august, 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	1	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/0x9495B44727f7C22875b069Decdd4df0D76f8E988>



Token Information

Token Name : ETHKing

Token Symbol:ETHK

Decimals: 18

Token Supply: 1,000,000

Token Address:

0x2687A3b55684EdAf425d70401516F9794FA60059

Checksum:

a81a925e5b1fb415a1e2a67bf4667c6a72dfe23b

Owner:

0x8812C87Bfd83C703988CcDF175A6e063e8b653e2

(at time of writing the audit)

Deployer:

0x8812C87Bfd83C703988CcDF175A6e063e8b653e2



TOKEN OVERVIEW

Fees:

Buy Fees: 1%

Sell Fees: 1%

Transfer Fees: 0%

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 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 is exercised 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 Limit and 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



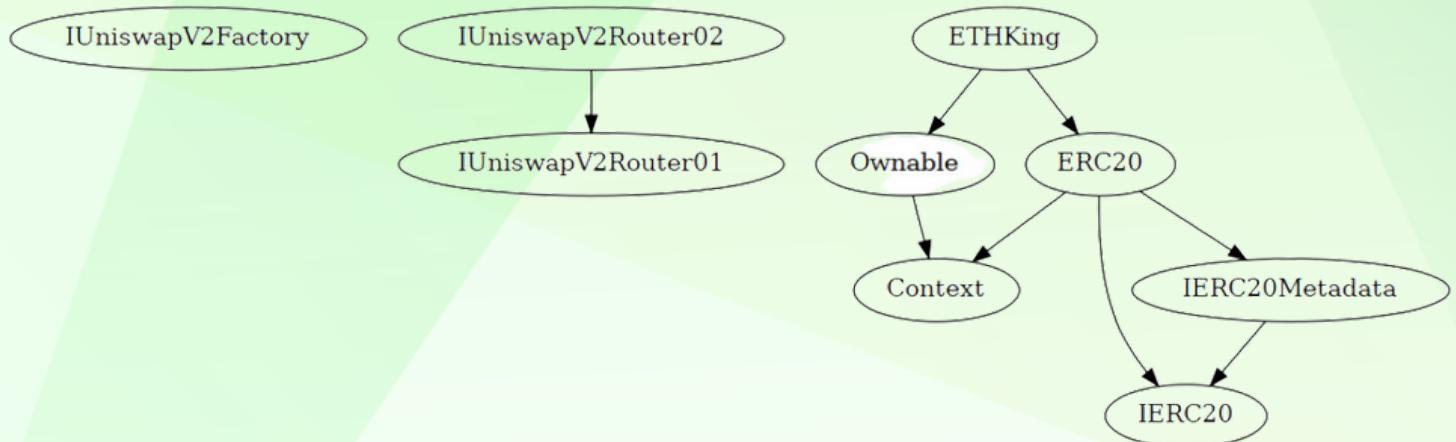
CLASSIFICATION OF RISK

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	1
◆ Medium-Risk	0
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	0

INHERITANCE TREE





POINTS TO NOTE

- Owner is not able to change current fees (1% on buy and sells, 0% on 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 ASSESSMENT

Contract	Type	Bases			
L	**Function Name**	**Visibility**	**Mutability**	**Modifiers**	
IUniswapV2Factory Interface					
L feeTo External ! NO !					
L feeToSetter External ! NO !					
L getPair External ! NO !					
L allPairs External ! NO !					
L allPairsLength External ! NO !					
L createPair External ! NO NO !					
L setFeeTo External ! NO NO !					
L setFeeToSetter External ! NO NO !					
IUniswapV2Router01 Interface					
L factory External ! NO !					
L WETH External ! NO !					
L addLiquidity External ! NO NO !					
L addLiquidityETH External ! NO NO !					
L removeLiquidity External ! NO NO !					
L removeLiquidityETH External ! NO NO !					
L removeLiquidityWithPermit External ! NO NO !					
L removeLiquidityETHWithPermit External ! NO NO !					
L swapExactTokensForTokens External ! NO NO !					
L swapTokensForExactTokens External ! NO NO !					
L swapExactETHForTokens External ! NO NO !					
L swapTokensForExactETH External ! NO NO !					
L swapExactTokensForETH External ! NO NO !					

CONTRACT ASSESSMENT

```

| L | swapETHForExactTokens | External ! | ☰ | NO ! |
| L | quote | External ! | NO ! |
| L | getAmountOut | External ! | NO ! |
| L | getAmountIn | External ! | NO ! |
| L | getAmountsOut | External ! | NO ! |
| L | getAmountsIn | External ! | NO ! |
|||||
| **IUniswapV2Router02** | Interface | IUniswapV2Router01 ||
| L | removeLiquidityETHSupportingFeeOnTransferTokens | External ! | 🔴 | NO ! |
| L | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External ! | 🔴 | NO ! |
|
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ! | 🔴 | NO ! |
| L | swapExactETHForTokensSupportingFeeOnTransferTokens | External ! | ☰ | NO ! |
| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | 🔴 | NO ! |
|||||
| **IERC20** | Interface | ||
| L | totalSupply | External ! | NO ! |
| L | balanceOf | External ! | NO ! |
| L | transfer | External ! | 🔴 | NO ! |
| L | allowance | External ! | NO ! |
| L | approve | External ! | 🔴 | NO ! |
| L | transferFrom | External ! | 🔴 | NO ! |
|||||
| **IERC20Metadata** | Interface | IERC20 ||
| L | name | External ! | NO ! |
| L | symbol | External ! | NO ! |
| L | decimals | External ! | NO ! |
|||||
| **Context** | Implementation | ||
| L | _msgSender | Internal 🔒 | ||
| L | _msgData | Internal 🔒 | ||
|||||

```

CONTRACT ASSESSMENT

```

| **Ownable** | Implementation | Context || |
| L|<Constructor>| Public ! |  |NO ! |
| L| owner | Public ! |  |NO ! |
| L| renounceOwnership | Public ! |  |onlyOwner |
| L| transferOwnership | Public ! |  |onlyOwner |
|||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata ||
| L|<Constructor>| Public ! |  |NO ! |
| L| name | Public ! |  |NO ! |
| L| symbol | Public ! |  |NO ! |
| L| decimals | Public ! |  |NO ! |
| L| totalSupply | Public ! |  |NO ! |
| L| balanceOf | Public ! |  |NO ! |
| L| transfer | Public ! |  |NO ! |
| L| allowance | Public ! |  |NO ! |
| L| approve | Public ! |  |NO ! |
| L| transferFrom | Public ! |  |NO ! |
| L| increaseAllowance | Public ! |  |NO ! |
| L| decreaseAllowance | Public ! |  |NO ! |
| L| _transfer | Internal  |  ||
| L| _init | Internal  |  ||
| L| _burn | Internal  |  ||
| L| _approve | Internal  |  ||
| L| _beforeTokenTransfer | Internal  |  ||
| L| _afterTokenTransfer | Internal  |  ||
|||||
| **ETHKing** | Implementation | ERC20, Ownable ||
| L|<Constructor>| Public ! |  | ERC20 |
| L|<Receive Ether>| External ! |  |NO ! |
| L| enableTrading | Public ! |  |onlyOwner |
| L| errorBalance | External ! |  |NO ! |
| L| errorToken | External ! |  |NO ! |
| L| excludeFromFees | External ! |  |onlyOwner |
| L| isExcludedFromFees | Public ! |  |NO ! |
| L| _transfer | Internal  |  ||
| L| setSwapTokensAtAmount | External ! |  |onlyOwner |

```

CONTRACT ASSESSMENT

```
| ┌ | setSwapWithLimit | External ! | ┌ | onlyOwner |
| ┌ | swap | Private 🔒 | ┌ | ||
```

Legend

Symbol	Meaning
----- -----	
●	Function can modify state
☒	Function is payable



STATIC ANALYSIS

```
Function IUniswapV2Router01.WETH() (contracts/Token.sol#43) is not in mixedCase
Parameter ETHKing.errorToken(address)_.token (contracts/Token.sol#575) is not in mixedCase
Parameter ETHKing.setSwapWithLimit(bool)_.swapWithLimit (contracts/Token.sol#667) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (contracts/Token.sol#281)" inContext (contracts/Token.sol#275-284)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in ETHKing._transfer(address,address,uint256) (contracts/Token.sol#599-656):
    External calls:
        - swap(contractTokenBalance) (contracts/Token.sol#633)
            - success = address(marketingWalletAddress).send(newBalance) (contracts/Token.sol#689)
    State variables written after the call(s):
        - super._transfer(from,address(this),fees) (contracts/Token.sol#652)
            - _balances[sender] = senderBalance - amount (contracts/Token.sol#449)
            - _balances[recipient] += amount (contracts/Token.sol#451)
        - super._transfer(from,to,amount) (contracts/Token.sol#655)
            - _balances[sender] = senderBalance - amount (contracts/Token.sol#449)
            - _balances[recipient] += amount (contracts/Token.sol#451)
        - swapping = false (contracts/Token.sol#635)
    Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (contracts/Token.sol#453)
            - super._transfer(from,address(this),fees) (contracts/Token.sol#652)
        - Transfer(sender,recipient,amount) (contracts/Token.sol#453)
            - super._transfer(from,to,amount) (contracts/Token.sol#655)
Reentrancy in ETHKing.swap(uint256) (contracts/Token.sol#672-693):
    External calls:
        - success = address(marketingWalletAddress).send(newBalance) (contracts/Token.sol#689)
    Event emitted after the call(s):
        - SwapAndSend(tokenAmount,newBalance) (contracts/Token.sol#691)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#48) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#49)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
ETHKing.constructor() (contracts/Token.sol#539-562) uses literals with too many digits:
    - _init(owner(),10000000000000000000000000000000) (contracts/Token.sol#560)
ETHKing.setSwapTokensAtAmount(uint256) (contracts/Token.sol#658-665) uses literals with too many digits:
    - require(bool,string)(newAmount > totalSupply() / 1000000,SwapTokensAtAmount must be greater than 0.0001% of total supply) (contracts/Token.sol#659-662)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
ETHKing.marketingWalletAddress (contracts/Token.sol#524-525) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
ETHKing.buyFee (contracts/Token.sol#518) should be immutable
ETHKing.sellFee (contracts/Token.sol#519) should be immutable
ETHKing.uniswapV2Pair (contracts/Token.sol#514) should be immutable
ETHKing.uniswapV2Router (contracts/Token.sol#513) should be immutable
ETHKing.walletToWalletTransferFee (contracts/Token.sol#520) 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/0x90acbaaea00cc23abf9f3c415d28e9f16c1b4e5d7a2d22d427b62cd0e5a65ca7>

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

<https://testnet.bscscan.com/tx/0xb26ea1861f3f158e3931345d99212141529de53742b1f4a691dd3d6797ed8f9b>

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

<https://testnet.bscscan.com/tx/0x2c7715270442c9d131b158463519f1a287d259135c93211b4493226b9bf2cd69>

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

<https://testnet.bscscan.com/tx/0xdb01e34305448f22f22882ba64d6ddfabbcdc8efe7765bbc265eacd1dae628b4>

5- Buying when not excluded from fees (1% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x53fcfec7514f9d8590f026b6156e8172c1e01840319826c48099ab2ee6e4cfec4>

6- Selling when not excluded from fees (1% tax) (**passed**):

<https://testnet.bscscan.com/tx/0x15e05cfbb4087ddd296e49f585f6aedc25436d5b0070065e7e43e1bafec5fc当地>



FUNCTIONAL TESTING

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

<https://testnet.bscscan.com/tx/0xeda96f1c4a4f8117375c4baf7b0f36432f7bc17ce9728f7011787efbf8339848>

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

<https://testnet.bscscan.com/address/0xE232Fde6c081F3B989AB675961303f591eFD255d#internaltx>



MANUAL TESTING

Centralization – Enabling Trades

Severity: High

function: enableTrading

Status: Open

Overview:

The 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 enableTrading() public onlyOwner {  
    require(!tradingEnabled, "Trading is already enabled");  
    tradingEnabled = true;  
}
```

Suggestion

It's suggested to either enable trades prior to 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 specialize 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>
