expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

USR

TOKEN OVERVIEW

Risk Findings

Severity	Found
🔴 High	6
🟠 Medium	1
🟡 Low	0
🔵 Informational	0

Centralization Risks

Owner Privileges	Description
🔴 Can Owner Set Taxes >25% ?	Detected
🟢 Owner needs to enable trading ?	Not Detected
🔴 Can Owner Disable Trades ?	Detected
🟢 Can Owner Mint ?	Not Detected
🔴 Can Owner Blacklist ?	Detected
🔴 Can Owner set Max Wallet amount ?	Detected
🟢 Can Owner Set Max TX amount ?	Not Detected

TABLE OF CONTENTS

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

Audit Result	Failed
KYC Verification	-
Audit Date	10 august 2023

CONTRACT DETAILS

Token Name: USR

Symbol: USR

Network: -

Language: -

Contract Address: -

Total Supply:420,000,000,000

Owner's Wallet: -

Deployer's Wallet: -

Testnet.
https://testnet.bscscan.com/token/0x3dDCD61164137d4F32A64d1e1Ad0c6B719041b82

AUDIT METHODOLOGY

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch , that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

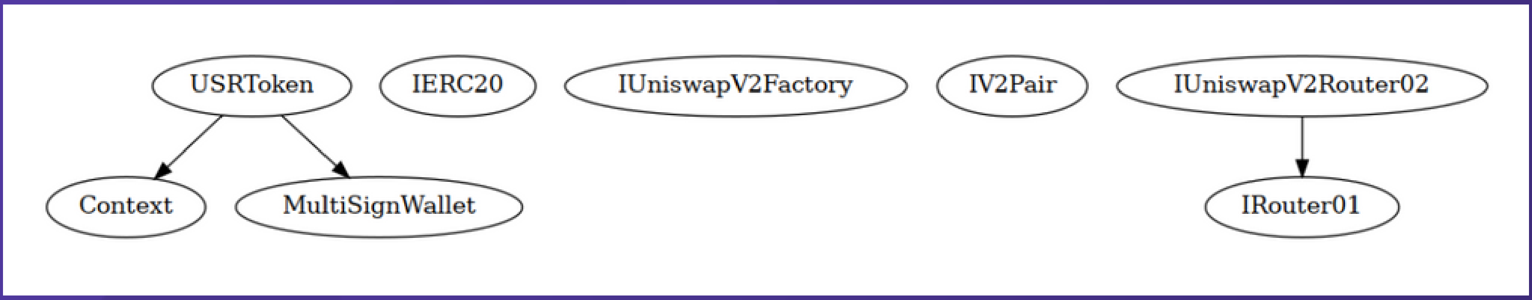
Low Risk

Issues on this level are minor details and warning that can remain unfixed.

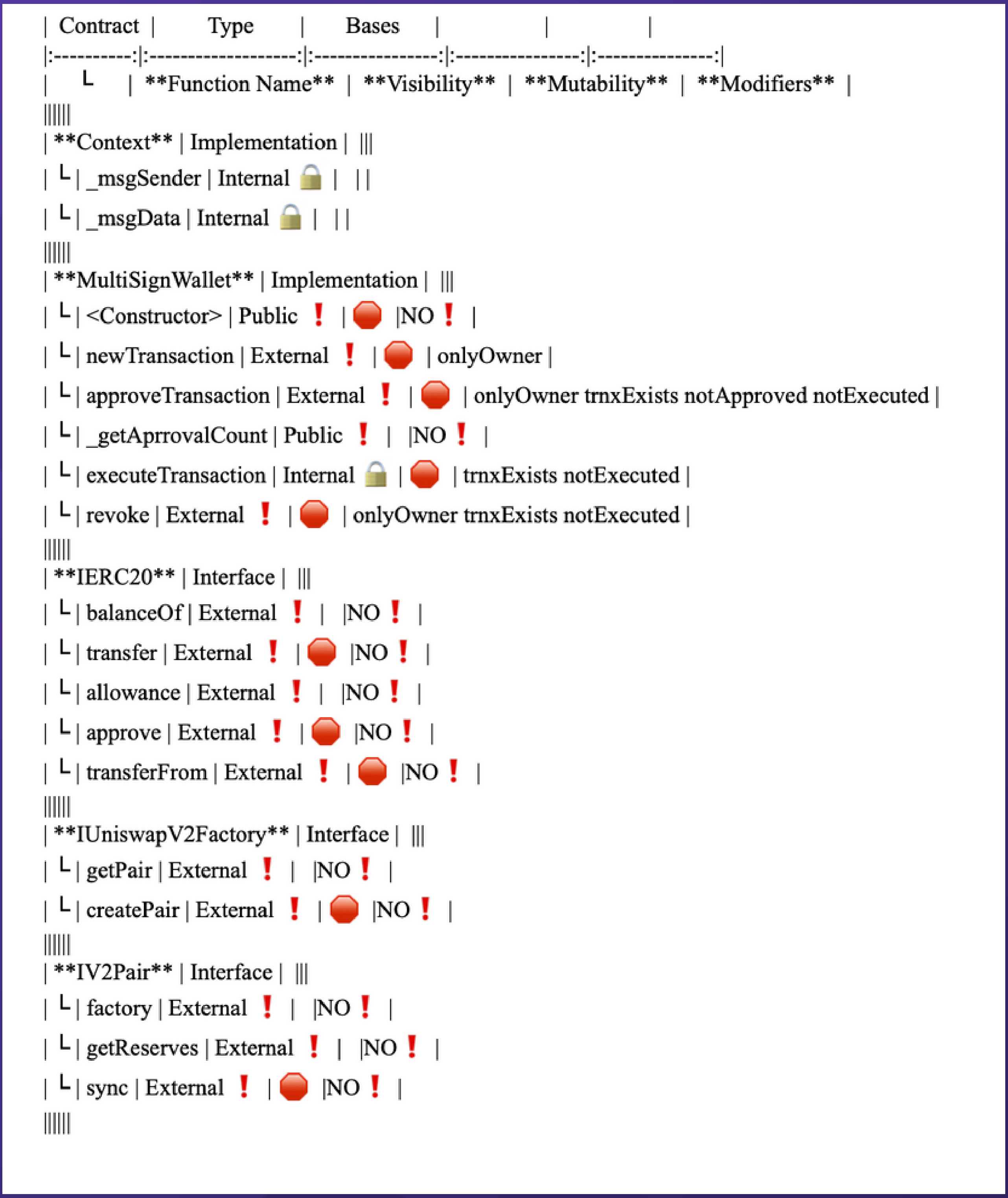
Informational

Issues on this level are minor details and warning that can remain unfixed.

INHERITANCE TREES

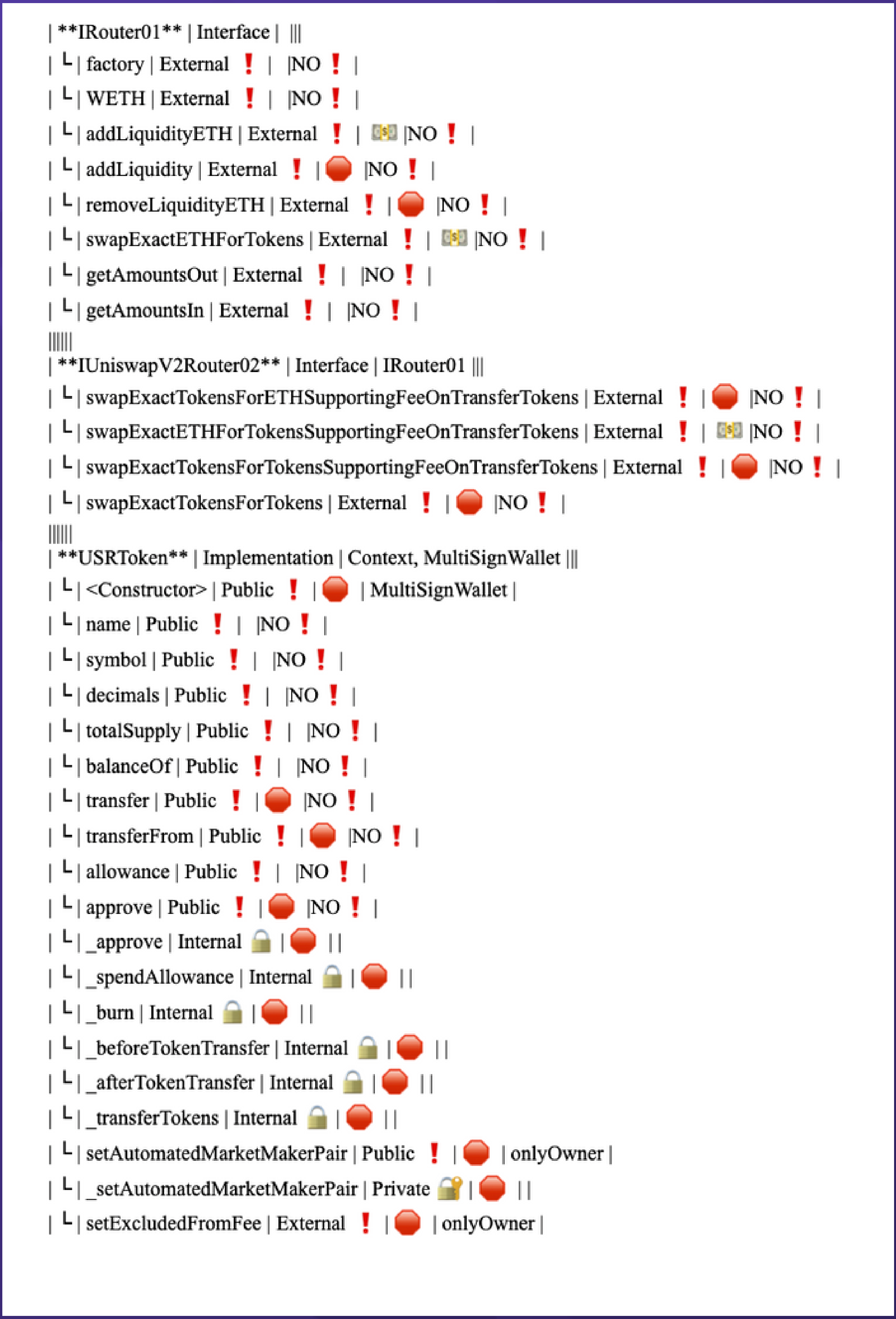


FUNCTION DETAILS



```
| Contract |        Type        |       Bases       |                  |                 |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
|    └    | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
||||||
| **Context** | Implementation | |||
| └ | _msgSender | Internal 🔒 | | |
| └ | _msgData | Internal 🔒 | | |
||||||
| **MultiSignWallet** | Implementation | |||
| └ | <Constructor> | Public ❗ | 🛑 |NO ❗ |
| └ | newTransaction | External ❗ | 🛑 | onlyOwner |
| └ | approveTransaction | External ❗ | 🛑 | onlyOwner trnxExists notApproved notExecuted |
| └ | _getAprrovalCount | Public ❗ | |NO ❗ |
| └ | executeTransaction | Internal 🔒 | 🛑 | trnxExists notExecuted |
| └ | revoke | External ❗ | 🛑 | onlyOwner trnxExists notExecuted |
||||||
| **IERC20** | Interface | |||
| └ | balanceOf | External ❗ | |NO ❗ |
| └ | transfer | External ❗ | 🛑 |NO ❗ |
| └ | allowance | External ❗ | |NO ❗ |
| └ | approve | External ❗ | 🛑 |NO ❗ |
| └ | transferFrom | External ❗ | 🛑 |NO ❗ |
||||||
| **IUniswapV2Factory** | Interface | |||
| └ | getPair | External ❗ | |NO ❗ |
| └ | createPair | External ❗ | 🛑 |NO ❗ |
||||||
| **IV2Pair** | Interface | |||
| └ | factory | External ❗ | |NO ❗ |
| └ | getReserves | External ❗ | |NO ❗ |
| └ | sync | External ❗ | 🛑 |NO ❗ |
||||||
```

FUNCTION DETAILS



```
| **IRouter01** | Interface |  |||
| └ | factory | External ❗️ |   |NO ❗️ |
| └ | WETH | External ❗️ |   |NO ❗️ |
| └ | addLiquidityETH | External ❗️ |  💵 |NO ❗️ |
| └ | addLiquidity | External ❗️ | 🛑  |NO ❗️ |
| └ | removeLiquidityETH | External ❗️ | 🛑  |NO ❗️ |
| └ | swapExactETHForTokens | External ❗️ |  💵 |NO ❗️ |
| └ | getAmountsOut | External ❗️ |   |NO ❗️ |
| └ | getAmountsIn | External ❗️ |   |NO ❗️ |
||||||
| **IUniswapV2Router02** | Interface | IRouter01 |||
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗️ | 🛑  |NO ❗️ |
| └ | swapExactETHForTokensSupportingFeeOnTransferTokens | External ❗️ |  💵 |NO ❗️ |
| └ | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ❗️ | 🛑  |NO ❗️ |
| └ | swapExactTokensForTokens | External ❗️ | 🛑  |NO ❗️ |
||||||
| **USRToken** | Implementation | Context, MultiSignWallet |||
| └ | <Constructor> | Public ❗️ | 🛑  | MultiSignWallet |
| └ | name | Public ❗️ |   |NO ❗️ |
| └ | symbol | Public ❗️ |   |NO ❗️ |
| └ | decimals | Public ❗️ |   |NO ❗️ |
| └ | totalSupply | Public ❗️ |   |NO ❗️ |
| └ | balanceOf | Public ❗️ |   |NO ❗️ |
| └ | transfer | Public ❗️ | 🛑  |NO ❗️ |
| └ | transferFrom | Public ❗️ | 🛑  |NO ❗️ |
| └ | allowance | Public ❗️ |   |NO ❗️ |
| └ | approve | Public ❗️ | 🛑  |NO ❗️ |
| └ | _approve | Internal 🔒 | 🛑  | |
| └ | _spendAllowance | Internal 🔒 | 🛑  | |
| └ | _burn | Internal 🔒 | 🛑  | |
| └ | _beforeTokenTransfer | Internal 🔒 | 🛑  | |
| └ | _afterTokenTransfer | Internal 🔒 | 🛑  | |
| └ | _transferTokens | Internal 🔒 | 🛑  | |
| └ | setAutomatedMarketMakerPair | Public ❗️ | 🛑  | onlyOwner |
| └ | _setAutomatedMarketMakerPair | Private 🔐 | 🛑  | |
| └ | setExcludedFromFee | External ❗️ | 🛑  | onlyOwner |
```

FUNCTION DETAILS

```
| └ | enableTrade | Public ❗ | 🛑 | onlyOwner |
| └ | pauseTrade | Public ❗ | 🛑 | onlyOwner |
| └ | disableBurn | Public ❗ | 🛑 | onlyOwner |
| └ | setUserFoundationWallet | External ❗ | 🛑 | onlyOwner |
| └ | updateShares | Internal 🔒 | 🛑 | |
| └ | setBuyUserFoundationPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setBuyLiquidityPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setBuyBurnPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setSellUserFoundationPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setSellLiquidityPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setSellBurnPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setTransferLiquidityPercentage | External ❗ | 🛑 | onlyOwner |
| └ | setTransferBurnPercentage | External ❗ | 🛑 | onlyOwner |
| └ | addToBlacklist | Public ❗ | 🛑 | onlyOwner |
| └ | removeFromBlacklist | Public ❗ | 🛑 | onlyOwner |
| └ | setTaxThreshold | External ❗ | 🛑 | onlyOwner |
| └ | setNumberOfBlocksForBlacklist | External ❗ | 🛑 | onlyOwner |
| └ | setMaxAmount | External ❗ | 🛑 | onlyOwner |
| └ | recoverTokensFromContract | External ❗ | 🛑 | onlyOwner |
| └ | recoverETHfromContract | External ❗ | 🛑 | onlyOwner |
| └ | recoverUSRfromUser | External ❗ | 🛑 | onlyOwner |
| └ | swapTokensForEth | Private 🔐 | 🛑 | |
| └ | swapAndLiquify | Internal 🔒 | 🛑 | |
| └ | addLiquidity | Private 🔐 | 🛑 | |
| └ | _transfer | Internal 🔒 | 🛑 | |
| └ | _calculateTax | Internal 🔒 | | |
| └ | <Fallback> | External ❗ | 💵 |NO ❗ |
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |


### Legend

| Symbol | Meaning |
|:--------:|-----------|
| 🛑 | Function can modify state |
| 💵 | Function is payable |
```

TESTNET VERSION

Adding Liquidity ✓
Tx:
https://testnet.bscscan.com/tx/0x4406f7c6d5499d8a8e7f8d8016af900f44b4120abc805d0f8ae6783d5d5f4855

==

Buying when excluded from fees ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0x05f38e26917e0adaaa5a3595f550a111bb9483a6a0176ffc0d410f89f4d3da66

==

Selling when excluded from fees ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0x1befa2467408290ae0b51a59125dc01b9c186cd30e7927f3594df5006e7db57e

==

Transferring when excluded from fees ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0x2c839826b51650b8772adef0db5a81d14f9aff7b55f0307659a7e784250c5843

==

Buying ✓
Tx (0-75% tax):
https://testnet.bscscan.com/tx/0x3d4164433ef080f35e3379c3c693dc32780ae42af93027aefdd9b32dc01e1b79

TESTNET VERSION

Selling ✓
Tx (0-75% tax):
https://testnet.bscscan.com/tx/0x5aa58144482964119f8a0b1c7f349291b327da992c849e5a4188263c26931e7e

===

Transferring ✓
Tx (0-50% tax):
https://testnet.bscscan.com/tx/0x2fc96db8613d672e5acdfd0ddce0fec3a9d2b8480b25eb1a0bfd8363a614eb21

===

Internal swap (BNB to marketing wallet | reward token to dividend tracker | reward distribution) ✓
Tx:
https://testnet.bscscan.com/tx/0x5aa58144482964119f8a0b1c7f349291b327da992c849e5a4188263c26931e7e

MANUAL REVIEW

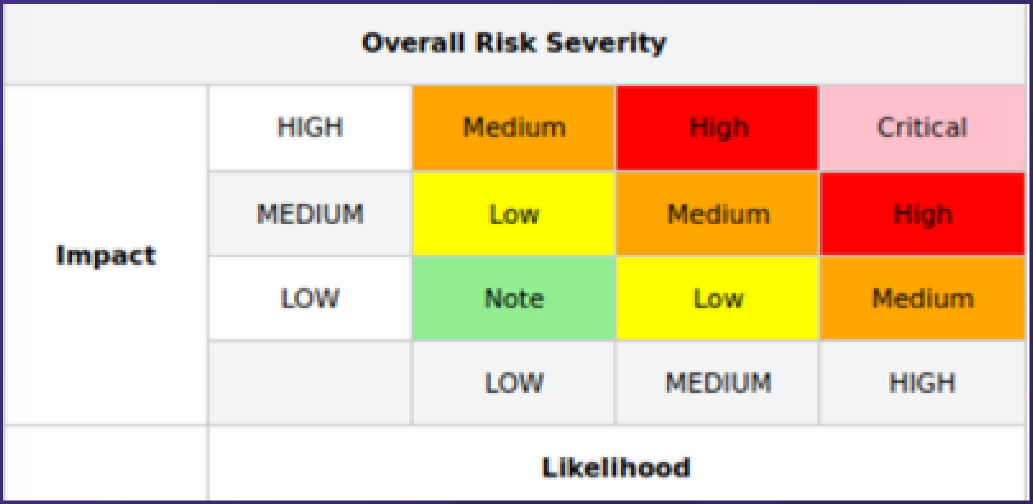
Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use



Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

HIGH RISK FINDING

Blacklisting

Category: Centralization
Status: Open
Impact: High

Overview:
Owner of the contract is able to blacklist an arbitrary address, blacklisted wallets are not able to buy/sell/transfer tokens.

```
 function addToBlacklist(address account) public onlyOwner {
   require(!blacklisted[account], "Account is already blacklisted");
   require(_msgSender() != account, "Cannot blacklist self");
   blacklisted[account] = true;
 }
```

Suggestion:
Delete addToBlacklist function and implement a more decentralized method for blacklisting bad actors such as MEV bots, sniper bots etc.

HIGH RISK FINDING

Excessive Fees

Category: Centralization
Status: Open
Impact: High

Overview:
Owner of the contract is able to set upto 75% tax on buys and 75% tax on sells as well as 50% tax on trasfers

```
function setBuyUserFoundationPercentage(...) external onlyOwner {...}
function setBuyLiquidityPercentage(...) external onlyOwner {...}
function setBuyBurnPercentage(...) external onlyOwner {...}
function setSellUserFoundationPercentage(...) external onlyOwner {...}
function setSellLiquidityPercentage(...) external onlyOwner {...}
function setSellBurnPercentage(...) external onlyOwner {...}
function setTransferLiquidityPercentage(...) external onlyOwner {...}
function setTransferBurnPercentage(...) external onlyOwner {...}
```

Suggestion:
75% is considered a very high amount of fee for investors. Hence its recommended to declare more reasonable upper bounds for buy/sell/transfer fees (e.g. 10% maximum tax for buy/sell/transfers)

HIGH RISK FINDING

Enabling/Disabling trades

Category: Centralization
Status: Open
Impact: High

Overview:
Trades are disabled by default owner of the contract is able enable/disable trades at anytime. When trades are disabled, no one would be able to transfer their tokens. Owner must enable trades In order to activate tokens tansfers.

```
 function pauseTrade() public onlyOwner {
   trade_open = false;
 }
 function disableBurn(bool _status) public onlyOwner {
   burn_disable = _status;
 }
```

Suggestion:
Ensure that trades remain enabled after enabling it

HIGH RISK FINDING

Maximum buy/sell/transfer

Category: Centralization
Status: Open
Impact: High

Overview:
Owner of the contract is able to limit buy/sell/transfer amount by a maximum limit. This limit can be set to any number including zero, setting this limit to zero will disable buy/sell/transfers for non-privilaged wallets

```
  function setMaxAmount(uint256 amount) external onlyOwner {
    maxAmount = amount;
  }
```

Suggestion:
Define a lower bound for maxAmount, e.g. maxAmount should not be less than 0.1% of total supply

```
  function setMaxAmount(uint256 amount) external onlyOwner {
 require(amount >= totalSupply() / 1000, “maximum amount must be
greater than 0.1% of totalsupply’);
    maxAmount = amount;
  }
```

HIGH RISK FINDING

Anti-bot can be renabled

Category: Centralization
Status: Open
Impact: High

Overview:
Owner of the contract is able to adjust dead blocks at anytime to any arbitrary number. Setting numBlocksForBlacklist to a large number can blacklist buyers even way after adding liquidity.

```
function setNumberOfBlocksForBlacklist(
  uint256 numBlocks
) external onlyOwner {
  numBlocksForBlacklist = numBlocks;
}
```

Suggestion:
Ensure that numBlocksForBlacklist is only adjustable in a reasonable range (0-5) blocks

HIGH RISK FINDING

Multisig contract is not implement in standard way

Category: Logical
Status: Open
Impact: High

Overview:
This MultiSignWallet contract implementation lacks a critical functionality of a multisig wallet: It does not define the actual transaction details to be executed upon approval by the required number of owners. The Transaction struct only contains a boolean isExecuted flag, with no details about destination address, value to be transferred, or data to be executed.
Furthermore, the executeTransaction function merely changes the isExecuted flag of the transaction, it does not actually call another contract or transfer any ETH or tokens, which would typically be expected in a multisig wallet implementation

Suggestion:
Its suggested to use a secure multisig wallet like Gnosis, however, to make current multisig contract functional you should :
- Expand the Transaction struct to include more details such as destination (the address to call or transfer funds to), value (the amount of Ether to transfer), and data (the function call data, if any). Here's an example:

```
struct Transaction {
 address destination;
 uint value;
 bytes data;
 bool isExecuted;
}
```

HIGH RISK FINDING

- In the newTransaction function, require these additional parameters and store them in the new transaction:

```
function newTransaction(address destination, uint value, bytes memory
data) external onlyOwner returns (uint256) {
 transactions.push(Transaction({
   destination: destination,
   value: value,
   data: data,
   isExecuted: false
 }));

 emit assignTrnx(transactions.length - 1);
 return transactions.length - 1;
}
```

Modify the executeTransaction function to use the low-level call function to actually perform the specified transaction, transferring the specified amount of Ether and calling a function if data is provided:

```
function executeTransaction(uint256 _trnxId) internal trnxExists(_trnxId)
notExecuted(_trnxId) {
 require(_getAprrovalCount(_trnxId) >= WalletRequired, "you don't have
sufficient approval");
 Transaction storage _transaction = transactions[_trnxId];
 (bool success, ) = _transaction.destination.call{value: _transaction.value}
(_transaction.data);
 require(success, "Execution failed.");
 _transaction.isExecuted = true;
 emit Execute(_trnxId);
}
```

These modifications would make the multisig wallet contract functional as expected, allowing owners to propose, approve, and execute arbitrary transactions with the funds controlled by the contract.

MEDIUM RISK FINDING

ILP tokens accumulated in the contract

Category: Centralisation
Status: Open
Impact: Medium
Overview:
Contract is receiving LP tokens generated form auto-liquidity. This LP tokens can be withdrawn by owner of the contract. LP tokens can be used to remove a portion of liquidity pool (both ETH and USR tokens)

```
 function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
   _approve(address(this), address(uniswapV2Router), tokenAmount);
   uniswapV2Router.addLiquidityETH{value: ethAmount}(
     address(this),
     tokenAmount,
     0, // slippage is unavoidable
     0, // slippage is unavoidable
     address(this),
     block.timestamp
   );
 }
```

Suggestion:
It is recommended to burn or lock new LP tokens.

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial	expelee
Expelee	expelee
expelee_official	expelee-co

Building the Futuristic Blockchain Ecosystem

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 guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee 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 Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway 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 Expelee team disclaims any liability for the resulting losses.

Building the Futuristic Blockchain Ecosystem