expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

TOKYO DOGE

TOKEN OVERVIEW

Risk Findings

Severity	Found
🔴 High	3
🟠 Medium	0
🟡 Low	0
🔵 Informational	0

Centralization Risks

Owner Privileges	Description
🟢 Can Owner Set Taxes >25% ?	No, fees cannot exceed 24%
🔴 Owner needs to enable trading ?	Yes, owner needs to enable trades
🟢 Can Owner Disable Trades ?	Not Detected
🟢 Can Owner Mint ?	Not Detected
🟢 Can Owner Blacklist ?	Not Detected
🟢 Can Owner set Max Wallet amount ?	Not 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	Passed
KYC Verification	No
Audit Date	5 June 2023

CONTRACT DETAILS

Token Name: Tokyo Doge

Symbol: Tokyodoge

Network: Binance smart chain

Language: Solidity

Decimals: 9

Token Type: BEP20

Checksum:
1373f22b8c5f4e279b234280ee3c1cc9c35b87d4

Contract Address: 0x22664443E849fc39F450f3F6996b159116834263

Total Supply: 10,000,000,000

Owner's Wallet: 0xd965B62C8dC4e20dCF55094b8d1c49bc74120fAa

Deployer's Wallet:
0xd965B62C8dC4e20dCF55094b8d1c49bc74120fAa

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.

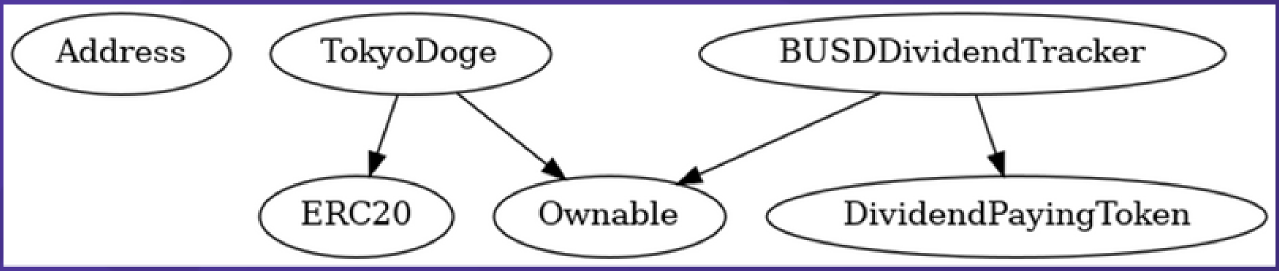
Low Risk

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

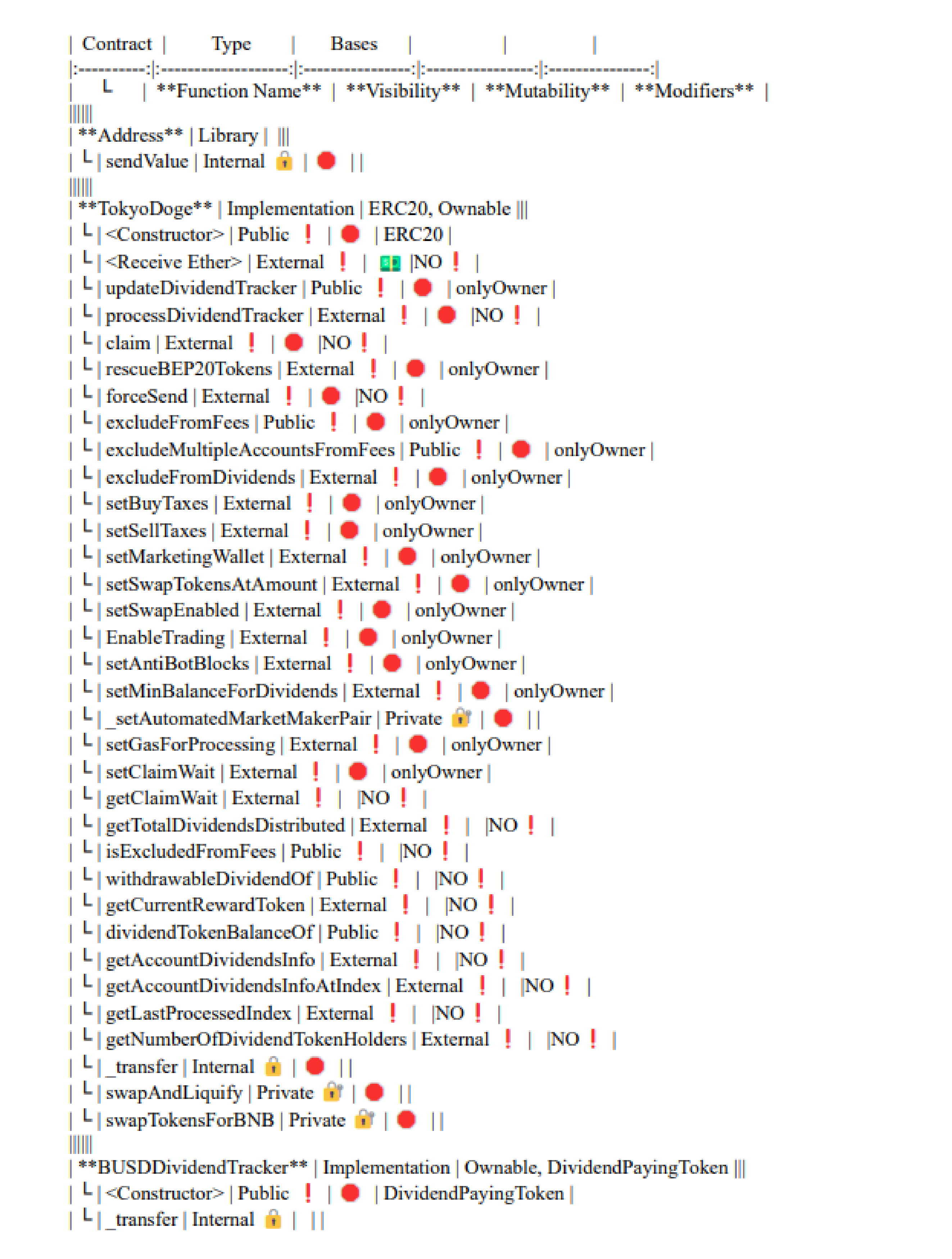
Informational

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

INHERITANCE TREES

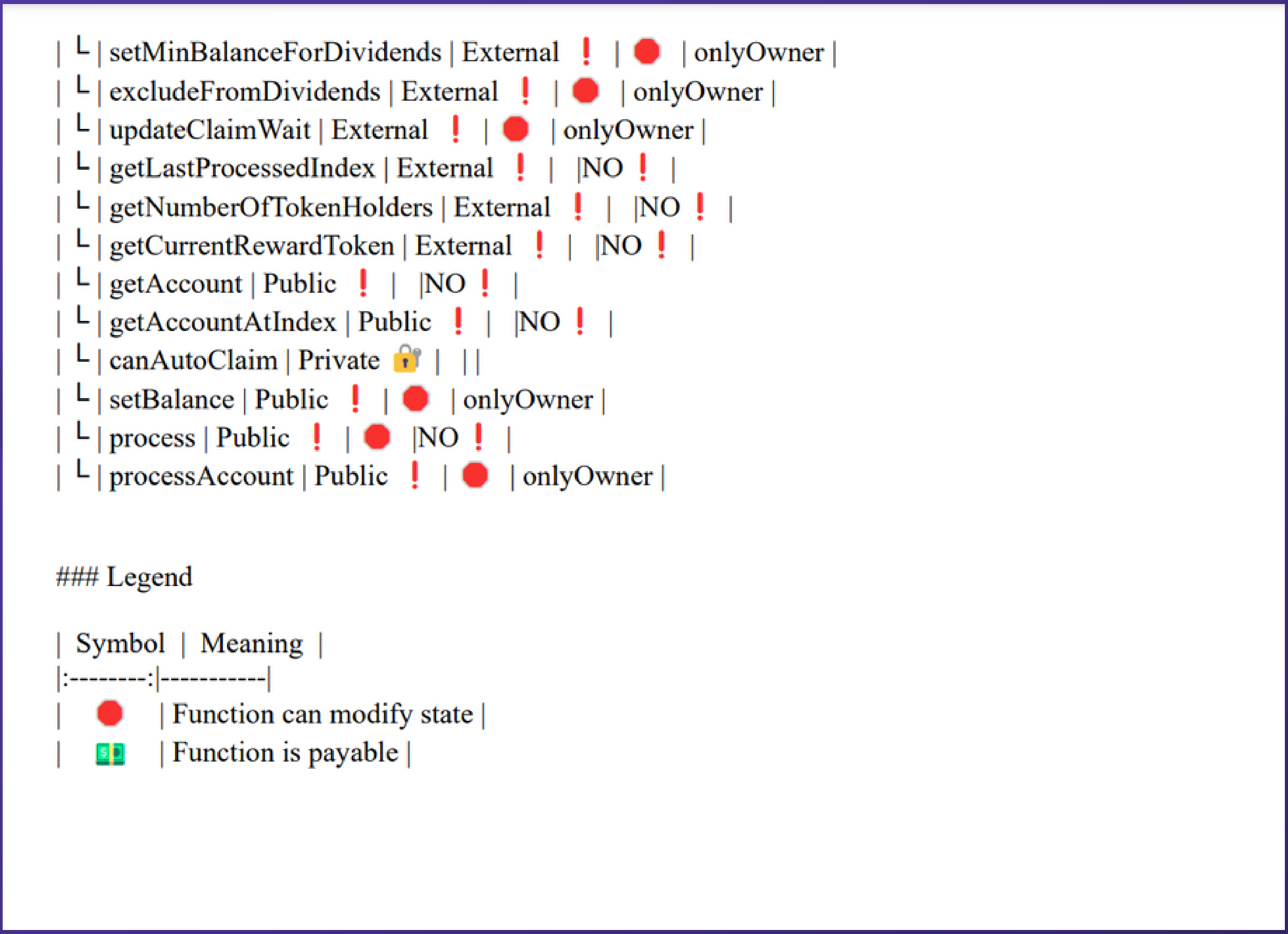


FUNCTION DETAILS



```
| Contract | Type | Bases | | |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
| └ | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
||||||
| **Address** | Library | |||
| └ | sendValue | Internal 🔒 | 🔴 | |
||||||
| **TokyoDoge** | Implementation | ERC20, Ownable |||
| └ | <Constructor> | Public ❗ | 🔴 | ERC20 |
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |
| └ | updateDividendTracker | Public ❗ | 🔴 | onlyOwner |
| └ | processDividendTracker | External ❗ | 🔴 |NO ❗ |
| └ | claim | External ❗ | 🔴 |NO ❗ |
| └ | rescueBEP20Tokens | External ❗ | 🔴 | onlyOwner |
| └ | forceSend | External ❗ | 🔴 |NO ❗ |
| └ | excludeFromFees | Public ❗ | 🔴 | onlyOwner |
| └ | excludeMultipleAccountsFromFees | Public ❗ | 🔴 | onlyOwner |
| └ | excludeFromDividends | External ❗ | 🔴 | onlyOwner |
| └ | setBuyTaxes | External ❗ | 🔴 | onlyOwner |
| └ | setSellTaxes | External ❗ | 🔴 | onlyOwner |
| └ | setMarketingWallet | External ❗ | 🔴 | onlyOwner |
| └ | setSwapTokensAtAmount | External ❗ | 🔴 | onlyOwner |
| └ | setSwapEnabled | External ❗ | 🔴 | onlyOwner |
| └ | EnableTrading | External ❗ | 🔴 | onlyOwner |
| └ | setAntiBotBlocks | External ❗ | 🔴 | onlyOwner |
| └ | setMinBalanceForDividends | External ❗ | 🔴 | onlyOwner |
| └ | _setAutomatedMarketMakerPair | Private 🔐 | 🔴 | |
| └ | setGasForProcessing | External ❗ | 🔴 | onlyOwner |
| └ | setClaimWait | External ❗ | 🔴 | onlyOwner |
| └ | getClaimWait | External ❗ | |NO ❗ |
| └ | getTotalDividendsDistributed | External ❗ | |NO ❗ |
| └ | isExcludedFromFees | Public ❗ | |NO ❗ |
| └ | withdrawableDividendOf | Public ❗ | |NO ❗ |
| └ | getCurrentRewardToken | External ❗ | |NO ❗ |
| └ | dividendTokenBalanceOf | Public ❗ | |NO ❗ |
| └ | getAccountDividendsInfo | External ❗ | |NO ❗ |
| └ | getAccountDividendsInfoAtIndex | External ❗ | |NO ❗ |
| └ | getLastProcessedIndex | External ❗ | |NO ❗ |
| └ | getNumberOfDividendTokenHolders | External ❗ | |NO ❗ |
| └ | _transfer | Internal 🔒 | 🔴 | |
| └ | swapAndLiquify | Private 🔐 | 🔴 | |
| └ | swapTokensForBNB | Private 🔐 | 🔴 | |
||||||
| **BUSDDividendTracker** | Implementation | Ownable, DividendPayingToken |||
| └ | <Constructor> | Public ❗ | 🔴 | DividendPayingToken |
| └ | _transfer | Internal 🔒 | | |
```

FUNCTION DETAILS



└	setMinBalanceForDividends	External ❗	🛑	onlyOwner
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO ❗
└	getNumberOfTokenHolders	External ❗		NO ❗
└	getCurrentRewardToken	External ❗		NO ❗
└	getAccount	Public ❗		NO ❗
└	getAccountAtIndex	Public ❗		NO ❗
└	canAutoClaim	Private 🔐		
└	setBalance	Public ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO ❗
└	processAccount	Public ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

TESTNET VERSION

Adding Liquidity ✓
Tx:
https://testnet.bscscan.com/tx/0x99ac92373d199984f679902eff8eba667b56049aeda53e4fa6e0d79ac3f66a41

==

Buying from a fee excluded wallet ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0xfb882c5f74b3d1c52f0652545b915b28f91ed01855bc27a48770eab96f4a8695

==

Selling from a fee excluded wallet ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0xfc8103158d124000fb585cc1e844014ed5570f31d4b6f110232856a615db43c2

==

Transferring using a fee excluded wallet ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0xebacf468e23ed2eaf367e7657938b6974112ff945e818e0d0c85e688ce0009bb

==

Buying from a regular wallet ✓
Tx (0-12% tax):
https://testnet.bscscan.com/tx/0x61d0eae4401a3fb16db013f879d9fae2742ebcdfcf1e5ef4f5ad24b12dce8f1e

==

Selling from a regular wallet ✓
Tx (0-12% tax):
https://testnet.bscscan.com/tx/0x697b6a798fe7c4f32e256bf9055e78d13df956d8f13997e1c8a1ce65c09e1a49

TESTNET VERSION

Transferring a regular wallet ✓
Tx (0%):
https://testnet.bscscan.com/tx/0x80e17c16f100deb70d3ee4c78e3909abc3d94d3df71fa13b0812d265d3746b92

==

Internal swap (marketing BNB + Auto-liquidity) ✓
Tx:
https://testnet.bscscan.com/tx/0xa44edd5d0d76c0652f0127272f745953bf26ebca2298005a28a67bc5f15a99ce

MANUAL REVIEW

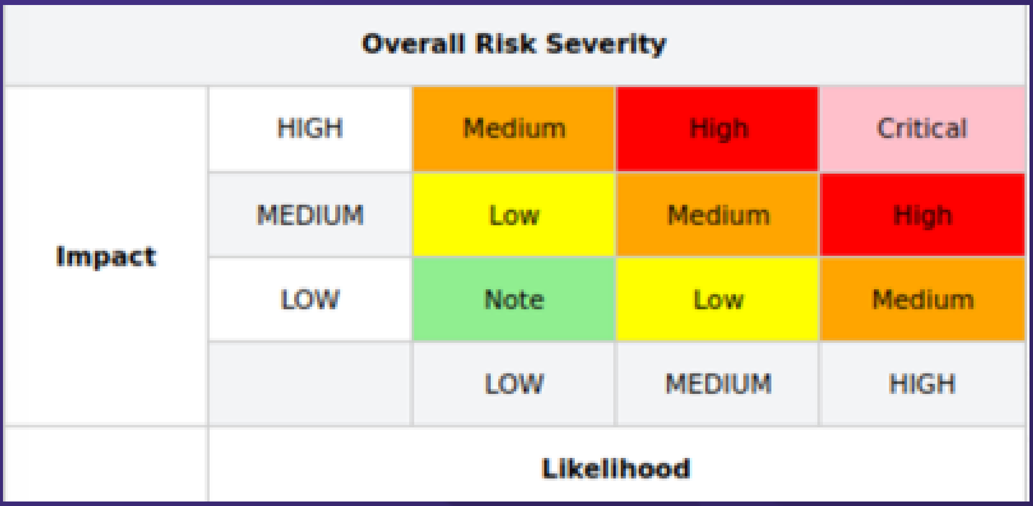
Severity Criteria

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

Setting internal swap threshold to 0

Severity : High
Category: Data validation
Status: Open

Overview

Setting swapTokensAtAmount to 0 can disable sell and transfers for non privileged wallets. This is because internal swap is performed even in case of swapTokensAtAmount being equal to 0.

```
  function setSwapTokensAtAmount(uint256 amount) external
onlyOwner {
      require(amount < totalSupply() / 100, "Swap Threshold should be
less than 1% of total supply");
      swapTokensAtAmount = amount;
  }
```

Suggestion:

to mitigate this issue, ensure that swapTokensAtAmount is always greater than a 0.

```
  function setSwapTokensAtAmount(uint256 amount) external
onlyOwner {
      require(amount < totalSupply() / 100, "Swap Threshold should be
less than 1% of total supply");
      require(amount > totalSupply() / 1000000, "Swap Threshold
should be more than .0001% of total supply");
      swapTokensAtAmount = amount;
  }
```

HIGH RISK FINDING

Enabling trades is not guaranteed

Severity : High
Category: Centralization

Status: Open

Overview
Owner must enable trades for investors manually. If trades remain disabled, holders wont be able to trade their tokens.

```
  function EnableTrading() external onlyOwner {
require(!tradingEnabled, "Trading is already enabled");
      tradingEnabled = true;
      startTradingBlock = block.number;
  }
```

Suggestion:

to mitigate this issue there are several options:
- Enable trades before end of presale
- Transfer ownership to a trusted 3rd party to guarantee enable of trades

HIGH RISK FINDING

Changing dividend tracker

Severity : High
Category: Logical

Status: Open

Overview

Owner is able to update dividend tracker. Setting dividend tracker to an invalid address can disable buy/sell/transfers for all holders (owner and whitelisted wallets included)

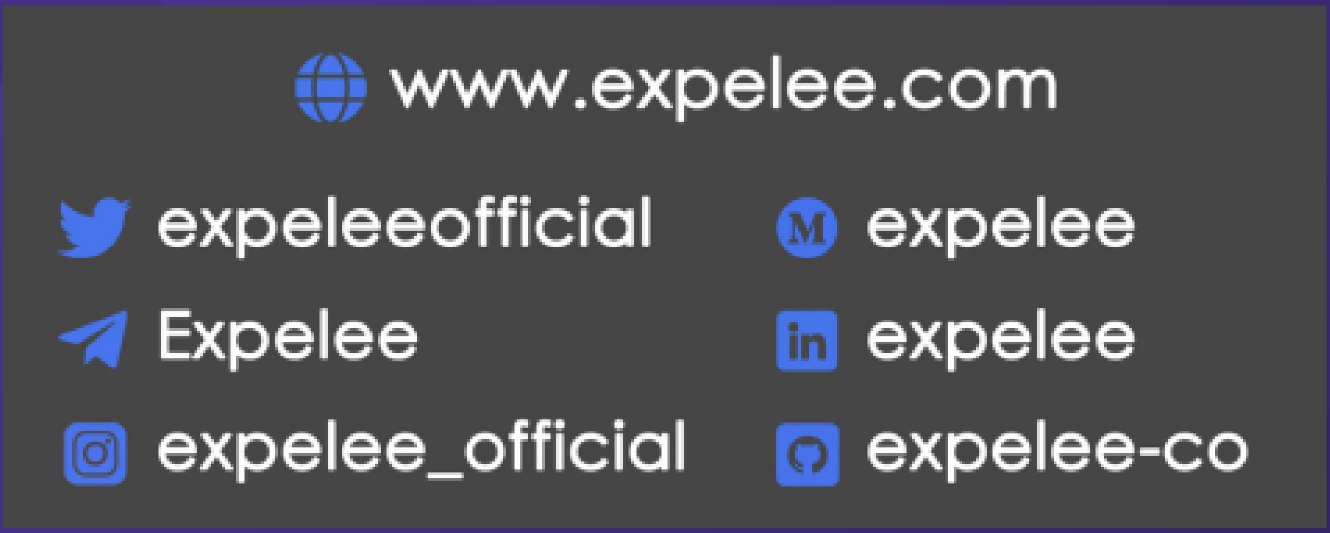
```
  function updateDividendTracker(address newAddress) public
onlyOwner {
      BUSDDividendTracker newDividendTracker =
BUSDDividendTracker(payable(newAddress));

newDividendTracker.excludeFromDividends(address(newDividendTr
acker), true);
newDividendTracker.excludeFromDividends(address(this), true);
newDividendTracker.excludeFromDividends(owner(), true);
newDividendTracker.excludeFromDividends(address(router), true);
      dividendTracker = newDividendTracker;
  }
```

Suggestion:

to mitigate this issue, ensure that dividend tracker is immutable, this can be achieved by deleting updateDividendTracker function.

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_official

expelee

expelee

expelee-co

expelee

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.

expelee

Building the Futuristic Blockchain Ecosystem