expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

OSCAR

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	1
Medium	0
Low	0
Informational	1

Centralization Risks

Owner Privileges	Description
Can Owner Set Taxes >25% ?	Not Detected
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	**-**
Audit Date	**17 June 2023**

CONTRACT DETAILS

Token Name: Oscar

Symbol: Oscar

Network: Ethereum

Language: Solidity

Contract Address: --

Total Supply: 50,000,000

Checksum:
84e5304fb6ba779760be7f36faed54162a40d89a

Owner's Wallet: --

Deployer's Wallet: --

Testnet.
https://testnet.bscscan.com/token/0x69D2Ec6E1E496c3e2eB5f160330e481463A7E52D

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.

expelee

INHERITANCE TREES

FUNCTION DETAILS

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
OscarToken	Implementation	IERC20, Ownable			
└	<Constructor>	Public ❗	🛑	NO ❗
└	<Receive Ether>	External ❗	💵	NO ❗
└	totalSupply	External ❗		NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	allowance	External ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	_approve	Internal 🔒	🛑	
└	approveMax	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
└	_transferFrom	Internal 🔒	🛑	
└	enableTrading	External ❗	🛑	onlyOwner
└	setAuthorizedWallets	External ❗	🛑	onlyOwner
└	rescueETH	External ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

TESTNET VERSION

Adding Liquidity ✅
Tx:
https://testnet.bscscan.com/tx/0xd7c54f41b28239945f72841e97464a36758bb57ed9d430452aa6063d16628bc4

==========

Buying ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0x984d739adba59d30db9676591dcb5b62dbbd024afa848d94a6f50fa03ae67208

==========

Selling ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0x399fa3c508eae4b5e933b601f5ae6d01452bf3c8498a935ce70756ef3364429f

==========

Transferring ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0xca54f4387fe5f03a497728b9efa35574e49c90a0b41f5697be402bec9ce069c0

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

Enabling trades is not guaranteed

Category: Centralization
Status: Open
Severity: High

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(!isTradeEnabled, "Trading already enabled");
    isTradeEnabled = true;
}
```

Suggestion:
There are multiple ways to resolve this issue:
•Enable trades prior to launch or presale : this ensurs that trades are already enabled forver and can not be disabled later. However, this completely negates the need for enableTrading function.
•Transfer Ownership of the contract to a trusted 3rd party: You can transfer ownership of the contract to a trusted 3rd party e.g a certified pinksale safu developer
•Create a time lock contract for enabling trades: this contract can enable trades after a fixed period of time.

INFORMATIONAL RISK FINDING

No way to withdraw stuck tokens

Category: Informational
Status: Open
Impact: Unknown (could be high depending on token)

Overview:

There are no functions to withdraw ERC20 tokens from the contract. If oscar tokens are sent to contract by mistake, those tokens would be locked forever.
The impact of this issue can vary depending on address of the token and amount of the tokens that were sent to the contract, hence, its highly recommended to have this function.

Suggestion:

Its suggeted to implement a function that enables owner to withdraw any ERC20 token from the contract. Since oscar token doesn't support Fees, oscar tokens can only be sent to the contract by mistake.

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

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.