expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

Miraverse

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	2
Medium	2
Low	0
Informational	1

Centralization Risks

Owner Privileges	Description
Can Owner Set Taxes >25% ?	Not Detected
Owner needs to enable trading ?	Not Detected
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

expelee

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 With High Risk**
KYC Verification	**Done**
Audit Date	**7 Aug 2023**

CONTRACT DETAILS

Token Name: Miraverse

Symbol: Mira

Network: BSC

Language: Solidity

Contract Address:
0x64293C84fC32f2A9B054fDA255cFE2F5fae64316

Total Supply: 1,000,000,000

Owner's Wallet:
0x66a95C92392d3eC3425648Cce52Ea6120e104Dbb

Deployer's Wallet:
0x66a95C92392d3eC3425648Cce52Ea6120e104Dbb

Testnet.
https://testnet.bscscan.com/address/0x7a2698DC1aCB8c8FC3E606D6347F1EE07091243A

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.

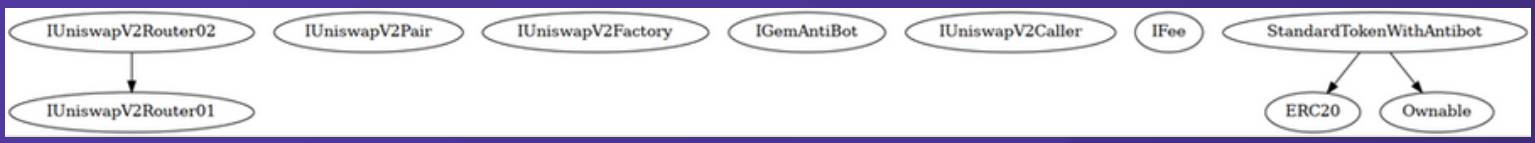
Low Risk

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



expelee

FUNCTION DETAILS

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IUniswapV2Router01	Interface			
└	factory	External ❗	NO ❗	
└	WETH	External ❗	NO ❗	
└	addLiquidity	External ❗	🔴	NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	removeLiquidity	External ❗	🔴	NO ❗
└	removeLiquidityETH	External ❗	🔴	NO ❗
└	removeLiquidityWithPermit	External ❗	🔴	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🔴	NO ❗
└	swapExactTokensForTokens	External ❗	🔴	NO ❗
└	swapTokensForExactTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokens	External ❗	💵	NO ❗
└	swapTokensForExactETH	External ❗	🔴	NO ❗
└	swapExactTokensForETH	External ❗	🔴	NO ❗
└	swapETHForExactTokens	External ❗	💵	NO ❗
└	quote	External ❗	NO ❗	
└	getAmountOut	External ❗	NO ❗	
└	getAmountIn	External ❗	NO ❗	
└	getAmountsOut	External ❗	NO ❗	
└	getAmountsIn	External ❗	NO ❗	
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
IUniswapV2Pair	Interface			
└	name	External ❗	NO ❗	
└	symbol	External ❗	NO ❗	
└	decimals	External ❗	NO ❗	
└	totalSupply	External ❗	NO ❗	
└	balanceOf	External ❗	NO ❗	
└	allowance	External ❗	NO ❗	

expelee

FUNCTION DETAILS

└	approve	External ❗	🔴	NO ❗
└	transfer	External ❗	🔴	NO ❗
└	transferFrom	External ❗	🔴	NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
└	PERMIT_TYPEHASH	External ❗		NO ❗
└	nonces	External ❗		NO ❗
└	permit	External ❗	🔴	NO ❗
└	MINIMUM_LIQUIDITY	External ❗		NO ❗
└	factory	External ❗		NO ❗
└	token0	External ❗		NO ❗
└	token1	External ❗		NO ❗
└	getReserves	External ❗		NO ❗
└	price0CumulativeLast	External ❗		NO ❗
└	price1CumulativeLast	External ❗		NO ❗
└	kLast	External ❗		NO ❗
└	mint	External ❗	🔴	NO ❗
└	burn	External ❗	🔴	NO ❗
└	swap	External ❗	🔴	NO ❗
└	skim	External ❗	🔴	NO ❗
└	sync	External ❗	🔴	NO ❗
└	initialize	External ❗	🔴	NO ❗
|||||
IUniswapV2Factory	Interface	|||
└	feeTo	External ❗		NO ❗
└	feeToSetter	External ❗		NO ❗
└	getPair	External ❗		NO ❗
└	allPairs	External ❗		NO ❗
└	allPairsLength	External ❗		NO ❗
└	createPair	External ❗	🔴	NO ❗
└	setFeeTo	External ❗	🔴	NO ❗
└	setFeeToSetter	External ❗	🔴	NO ❗
└	INIT_CODE_PAIR_HASH	External ❗		NO ❗
|||||
IGemAntiBot	Interface	|||
└	setTokenOwner	External ❗	🔴	NO ❗
└	onPreTransferCheck	External ❗	🔴	NO ❗
|||||
IUniswapV2Caller	Interface	|||
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
|||||
IFee	Interface	|||
└	payFee	External ❗	💶	NO ❗
|||||
StandardTokenWithAntibot	Implementation	ERC20, Ownable |||
└	<Constructor>	Public ❗	💶	ERC20
└	decimals	Public ❗		NO ❗

FUNCTION DETAILS

└	setUsingAntiBot	External ❗	🔴	onlyOwner
└	updateUniswapV2Pair	External ❗	🔴	onlyOwner
└	updateUniswapV2Router	Public ❗	🔴	onlyOwner
└	updateMaxWallet	External ❗	🔴	onlyOwner
└	updateMaxTransactionAmount	External ❗	🔴	onlyOwner
└	updateLiquidityFee	External ❗	🔴	onlyOwner
└	updateMarketingFee	External ❗	🔴	onlyOwner
└	updateMarketingWallet	External ❗	🔴	onlyOwner
└	updateMinAmountToTakeFee	External ❗	🔴	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🔴	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🔴	
└	excludeFromFee	External ❗	🔴	onlyOwner
└	excludeFromMaxTransactionAmount	External ❗	🔴	onlyOwner
└	_transfer	Internal 🔒	🔴	
└	takeFee	Private 🔐	🔴	lockTheSwap
└	swapTokensForBaseToken	Private 🔐	🔴	
└	addLiquidity	Private 🔐	🔴	
└	withdrawETH	External ❗	🔴	onlyOwner
└	withdrawToken	External ❗	🔴	onlyOwner
└	<Receive Ether>	External ❗	💵	NO ❗

Legend

Symbol	Meaning
🔴	Function can modify state
💵	Function is payable

TESTNET VERSION

Adding Liquidity ✅
Tx:
https://testnet.bscscan.com/tx/0x6d7d08e8b8703988eecf1056b05f6a93d672f4316cbb73f86493a9401054dcf2

==========

Buying when excluded from fees ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0x092b26afa4c273fffce6477b658d113153cadf78cf99c9e912bae8e593bf1ed5

==========

Selling when excluded from fees ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0x45f9296c302e14c9c2a937a64f19ea98b015067da17569cfce5487a4a571463b

==========

Transferring when excluded from fees ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0x88968cb22e8fa545f72675f61b59794b68850dac05ce960580ea2cf87c91a589

==========

Buying ✅
Tx (0-20% tax):
https://testnet.bscscan.com/tx/0xc3a0d499870f207fb9dcea05ff7bc480bea8b067e20a233868c733712cb201cf

TESTNET VERSION

Selling ✅
Tx (0-20% tax):
https://testnet.bscscan.com/tx/0xf25d050a5209363e281fb49ef10e8555408f0873c33502216fbb0be62219084f

==========

Transferring ✅
Tx (0% tax):
https://testnet.bscscan.com/tx/0x251f9245f57e3fd216fd5379a81dcea5aa3d9a96eb0a9e68c37ce72a276cdb97

==========

Internal swap (auto-liquidity + ETH sent to marketing wallet) ✅
Tx:
https://testnet.bscscan.com/tx/0xf25d050a5209363e281fb49ef10e8555408f0873c33502216fbb0be62219084f

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

updating uniswapV2 router

Category: Logical / centralization
Status: Open
Impact: High

Overview:

Owner is able to update mainRouter which is used for performing internal swap (auto liquidity and swapping fees into BNB). Setting mainRouter to a malicious contract could potentially disable internal swaps which in result affects sell transactions.

```
  function updateUniswapV2Router(address
newAddress) public onlyOwner {
      require(
          newAddress != address(mainRouter),
          "The router already has that address"
      );
      emit UpdateUniswapV2Router(newAddress,
address(mainRouter));
      mainRouter = IUniswapV2Router02(newAddress);
_approve(address(this), address(mainRouter), MAX);
```

HIGH RISK FINDING

```
 if (baseTokenForPair != mainRouter.WETH()) {
IERC20(baseTokenForPair).approve(address(mainRoute
r), MAX);
 }
 address _mainPair =
IUniswapV2Factory(mainRouter.factory()).createPair(
 address(this),
 baseTokenForPair
 );
 mainPair = _mainPair;
_setAutomatedMarketMakerPair(mainPair, true);
 }
```

Suggestion:

Its suggested to keep mainRouter immutable or disable updateUniswapV2Pair function after adding liquidity.

HIGH RISK FINDING

MaxTransaction amount not checked for liquidity pool

Category: Logical / centralization
Status: Open
Impact: High

Overview:

Below condition in _transfer function is not checking whether to is the pair address or not , in a sell condition, “to” is equal to pair address. If balance of liquidity pair is more than maxWallet and pair is not excluded form MaxTransactionAmount, this condition could cause sell transactions to revert

```
 if (!isExcludedFromMaxTransactionAmount[to]) {
      require(
          balanceOf(to) < maxWallet,
           "ERC20: exceeds max wallet limit"
      );
}
```

HIGH RISK FINDING

Suggestion:

check if “to” is not equal to pair address

```
 if (!isExcludedFromMaxTransactionAmount[to] &&
!automatedMarketMakerPairs[to]
) {
 require(
 balanceOf(to) < maxWallet,
 "ERC20: exceeds max wallet limit"
 );
}
```

MEDIUM RISK FINDING

up to 20% fee

Category: Centralization
Status: Open
Impact: Medium

Overview:

Owner is able to update buy/sell fees within 0-20 percent. The upper bound might be considered a high amount of tax for investors.

```
  function updateLiquidityFee(
      uint16 _sellLiquidityFee,
      uint16 _buyLiquidityFee
  ) external onlyOwner {
      require(
_sellLiquidityFee + (sellMarketingFee) <= 200,
          "sell fee <= 20%"
      );
require(_buyLiquidityFee + (buyMarketingFee) <= 200, "buy
fee <= 20%");
      emit UpdateLiquidityFee(
_sellLiquidityFee,
          _buyLiquidityFee,
```

MEDIUM RISK FINDING

```
sellLiquidityFee,
 buyLiquidityFee
 );
 sellLiquidityFee = _sellLiquidityFee;
 buyLiquidityFee = _buyLiquidityFee;
 }
```

Suggestion:

Its suggested to declare a more reasonable upper bound for fees. This upper bound is suggested to be 10% according to pinksale safu criteria

MEDIUM RISK FINDING

Invalid condition for updating maximum buy/sell/transfer

Category: Logical
Status: Open
Impact: Medium

Overview:

require statement of updateMaxTransactionAmount function is indicating that new maximum transaction amount should be less than 0.1% of total supply, but the condition is not considering dividing total supply by 1000.

```
  function updateMaxTransactionAmount(
      uint256 _maxTransactionAmount
  ) external onlyOwner {
      require(
_maxTransactionAmount >= totalSupply(),
"maxTransactionAmount >= total supply / 1000"
      );
      emit UpdateMaxTransactionAmount(
_maxTransactionAmount,
maxTransactionAmount
      );
      maxTransactionAmount = _maxTransactionAmount;
  }
```

MEDIUM RISK FINDING

Suggestion:

change require statement to match error message:

```
    require(
        _maxTransactionAmount >= totalSupply() / 1000,
"maxTransactionAmount >= total supply / 1000"
    );
```

INFORMATIONAL FINDING

Maximum wallet and buy/sell/transfer

Category: Centralization
Status: Open
Impact: Informational

Overview:

Owner is able to set maximum wallet and buy/sell/transfer. This limits have a lower bound of 0.1% of total supply.

```
  function updateMaxWallet(uint256 _maxWallet) external
onlyOwner {
      require(
          _maxWallet >= totalSupply() / 1000,
          "maxWallet >= total supply / 1000"
      );
      emit UpdateMaxWallet(_maxWallet, maxWallet);
      maxWallet = _maxWallet;
  }

  function updateMaxTransactionAmount(
      uint256 _maxTransactionAmount
  ) external onlyOwner {
      require(
_maxTransactionAmount >= totalSupply(),
```

INFORMATIONAL FINDING

```
"maxTransactionAmount >= total supply / 1000"
 );
 emit UpdateMaxTransactionAmount(
 _maxTransactionAmount,
maxTransactionAmount
 );
 maxTransactionAmount = _maxTransactionAmount;
 }
```

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.