expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

JEW

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	1
Medium	3
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

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	**19 June 2023**

CONTRACT DETAILS

Token Name: Happy Merchant

Symbol: JEW

Network: Binance smart chain

Language: Solidity

Contract Address:
0xbF56e97DAbB8F6b1b0Ef1a07D1C4D9C63851381c

Total Supply: 18,000,000

Owner's Wallet:
0x02a54094F727D1D83788e70C68937bde195A480C

Deployer's Wallet:
0x02a54094F727D1D83788e70C68937bde195A480C

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.

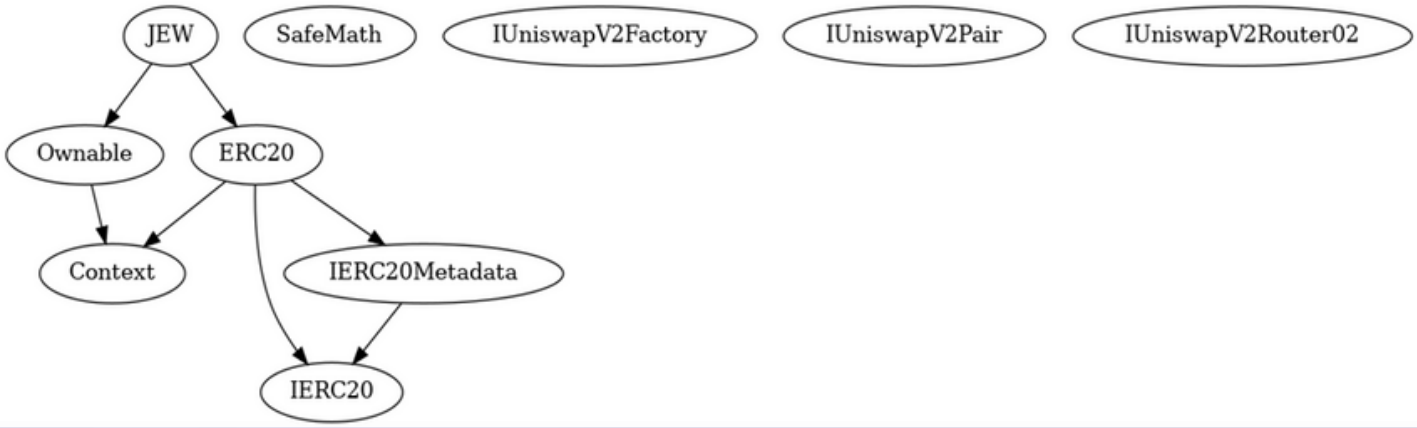
Low Risk

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



TESTNET VERSION

Adding Liquidity ✔
Tx:
https://testnet.bscscan.com/tx/0x0a52d7e3089d8e42fb730f34454b3737b0e027d61afb1aff84f9cb93d1a2fd84

==========

Buying from a fee excluded wallet ✔
Tx (0% tax):
https://testnet.bscscan.com/tx/0xf06b42588e003b54d6e0d301289b9a7a6c2ad27535d8da30ef031d20ca258661

==========

Selling from a fee excluded wallet ✔
Tx (0% tax):
https://testnet.bscscan.com/tx/0xa75614238fb7ccc44a7613167d92cacc0b601db0d0c6f448e33ef974778e332d

==========

Transferring using a fee excluded wallet ✔
Tx (0% tax):
https://testnet.bscscan.com/tx/0x8be58f85c2f9696c0477e5fe17c9bffc448d130b4a3aa3e019b4bb74408d0011

==========

TESTNET VERSION

Buying from a regular wallet ✓
Tx (0-20% tax):
https://testnet.bscscan.com/tx/0xdcb90c9fc0f9f569f4156b8f0e893b23eeba3cffce97a871e69973b67d204e3e

==========

Selling from a regular wallet ✓
Tx (0-20% tax):
https://testnet.bscscan.com/tx/0xb6171f818b16d82d698334ec34ed475b32cd5979407080bce187e0dbc1578028

==========

Transferring from a regular wallet ✓
Tx (0% tax):
https://testnet.bscscan.com/tx/0x79540daff4cd4a3ce08ae8cd659b7ad7a09bcc00dfe8f015e16e3bc4859f8975

==========

Internal swap (Auto-liquidity / Marketing and development wallets received BNB) ✓
Tx:
https://testnet.bscscan.com/tx/0xb6171f818b16d82d698334ec34ed475b32cd5979407080bce187e0dbc1578028

expelee

FUNCTION DETAILS

Contract	Type	Bases			
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**	
Context	Implementation				
└	_msgSender	Internal 🔒			
└	_msgData	Internal 🔒			
Ownable	Implementation	Context			
└	<Constructor>	Public ❗	🛑	NO ❗	
└	owner	Public ❗		NO ❗	
└	renounceOwnership	Public ❗	🛑	onlyOwner	
└	transferOwnership	Public ❗	🛑	onlyOwner	
└	_transferOwnership	Internal 🔒	🛑		
IERC20	Interface				
└	totalSupply	External ❗		NO ❗	
└	balanceOf	External ❗		NO ❗	
└	transfer	External ❗	🛑	NO ❗	
└	allowance	External ❗		NO ❗	
└	approve	External ❗	🛑	NO ❗	
└	transferFrom	External ❗	🛑	NO ❗	
IERC20Metadata	Interface	IERC20			
└	name	External ❗		NO ❗	
└	symbol	External ❗		NO ❗	
└	decimals	External ❗		NO ❗	
ERC20	Implementation	Context, IERC20, IERC20Metadata			
└	<Constructor>	Public ❗	🛑	NO ❗	
└	name	Public ❗		NO ❗	
└	symbol	Public ❗		NO ❗	
└	decimals	Public ❗		NO ❗	

expelee

FUNCTION DETAILS

└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	

SafeMath	Library	
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		

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 ❗

IUniswapV2Pair	Interface	
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗

expelee

FUNCTION DETAILS

└	allowance	External ❗			NO ❗	
└	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 ❗	

IUniswapV2Router02	Interface		|||
└	factory	External ❗		|NO ❗	
└	WETH	External ❗		|NO ❗	
└	addLiquidity	External ❗	🔴	NO ❗	
└	addLiquidityETH	External ❗	💵	NO ❗	
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗	
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗	
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗	

JEW	Implementation	ERC20, Ownable			
└	<Constructor>	Public ❗	🔴	ERC20	
└	<Receive Ether>	External ❗	💵	NO ❗	
└	enableTrading	External ❗	🔴	onlyOwner	
└	removeLimits	External ❗	🔴	onlyOwner	
└	disableTransferDelay	External ❗	🔴	onlyOwner	
└	updateSwapTokensAtAmount	External ❗	🔴	onlyOwner	
└	updateMaxTxnAmount	External ❗	🔴	onlyOwner	
└	updateMaxWalletAmount	External ❗	🔴	onlyOwner	
└	excludeFromMaxTransaction	Public ❗	🔴	onlyOwner	
└	updateSwapEnabled	External ❗	🔴	onlyOwner	
└	updateBuyFees	External ❗	🔴	onlyOwner	
└	updateSellFees	External ❗	🔴	onlyOwner	
└	excludeFromFees	Public ❗	🔴	onlyOwner	

FUNCTION DETAILS

└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateMarketingWalletInfo	External ❗	🛑	onlyOwner
└	updateDevelopmentWalletInfo	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗	|NO ❗	
└	_transfer	Internal 🔒	🛑	
└	swapTokensForEth	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	swapBack	Private 🔐	🛑	
└	setAutoLPBurnSettings	External ❗	🛑	onlyOwner
└	autoBurnLiquidityPairTokens	Internal 🔒	🛑	
└	manualBurnLiquidityPairTokens	External ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

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

Category: Numerical
Subject: Overflow at auto burn function
Status: Open
Severity : High

Overview

at setAutoLPBurnSettings function, owner is able to set lpBurnFrequency (time between burns) to any number greater than 600. setting lpBurnFrequency to uint256 max causes below condition to revert sell transactions:

```
    if (
        !swapping && automatedMarketMakerPairs[to] && lpBurnEnabled
            && block.timestamp >= lastLpBurnTime + lpBurnFrequency &&
!_isExcludedFromFees[from]
    ) {
autoBurnLiquidityPairTokens();
    }
```

Suggestion

Make sure that lpBurnFrequency is always less than a reasonable value (e.g. 10 days)

```
  function setAutoLPBurnSettings(uint256 _frequencyInSeconds, uint256 _percent,
bool _Enabled) external onlyOwner {
require(_frequencyInSeconds >= 600, "cannot set buyback more often than every 10
minutes");
      require(_percent <= 1000 && _percent >= 0, "Must set auto LP burn percent
between 0% and 10%");
      require(_ lpBurnFrequency <= 10 days, "cannot set buyback more than 10 days");
      lpBurnFrequency = _frequencyInSeconds;
      percentForLPBurn = _percent;
      lpBurnEnabled = _Enabled;
  }
```

HIGH RISK FINDING

Category: Centralization
Subject: Excessive fees
Status: Open
Severity : Medium

Overview

Owner is able to set up to 20% tax for buy and sells seperatly

```
function updateBuyFees(uint256 _marketingFee, uint256 _liquidityFee, uint256 _developmentFee) external onlyOwner {
    buyMarketingFee = _marketingFee;
    buyLiquidityFee = _liquidityFee;
    buyDevelopmentFee = _developmentFee;
    buyTotalFees = buyMarketingFee + buyLiquidityFee + buyDevelopmentFee;
    require(buyTotalFees <= 20, "Must keep fees at 35% or less");
}

function updateSellFees(uint256 _marketingFee, uint256 _liquidityFee, uint256 _developmentFee) external onlyOwner {
    sellMarketingFee = _marketingFee;
    sellLiquidityFee = _liquidityFee;
    sellDevelopmentFee = _developmentFee;
    sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevelopmentFee;
require(sellTotalFees <= 20, "Must keep fees at 40% or less");
}
```

Suggestion

Ensure that buy / sell fees are less than a reasonable value (10% suggested by pinksale safu criteria)
0 <= tota buy fees <= 10
0 <= tota sell fees <= 10
0 <= tota transfer fees <= 10

HIGH RISK FINDING

Category: Centralization
Subject: Limits
Status: Open
Severity : Medium

Overview

Owner is able to set max wallet/transfer/sell/buy amounts. This limits can not be less than 0.1% of total supply (for transfer/sell/buy maximum allowed amount) and less than 0.5% (for maximum balance of wallets).

```
function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
    require(newNum >= ((totalSupply() * 1) / 1000) / 1e18, "Cannot set
maxTransactionAmount lower than 0.1%");
    maxTransactionAmount = newNum * (10 ** 18);
}

function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
    require(newNum >= ((totalSupply() * 5) / 1000) / 1e18, "Cannot set maxWallet
lower than 0.5%");
    maxWallet = newNum * (10 ** 18);
}
```

Suggestion

According to pinksale safu criteria, its suggested to keep wallet limit always more than 1% of total supply.
1% of total supply <= max wallet

HIGH RISK FINDING

Category: : Missing Logic
Subject: Stuck Tokens and ETH
Status: Open
Severity : Medium

Overview

There are no function in the contract to be able to withdraw Stuck ETH or ERC20 tokens from the contract .

Suggestion

Implement a method to be able to withdraw stuck ERC20 and ETH from the contract by owner

HIGH RISK FINDING

Category: : Informational
Subject: Burning LP tokens
Status: Open
Severity : Unkonwn

Overview

The owner's ability to manually burn up to 10% of JEW tokens from the liquidity pool every 30 minutes, in conjunction with the auto-burn mechanism that can also burn up to 10% of JEW tokens from the liquidity pool (with a minimum interval of 10 minutes), could potentially lead to substantial fluctuations in the price of the token. High volatility in token prices may deter some investors or users due to increased unpredictability and risk.

Suggestion

It might be advisable to revise these mechanisms to balance token burn rates with the need for price stability. Here are a few recommendations:

1. Review the burn percentage: Reducing the maximum allowable burn rate could decrease potential price volatility. Instead of allowing up to 10% of tokens to be burned, consider a lower percentage.

2.Extend the burn interval: Increasing the time intervals between manual and auto burns could also help limit rapid price fluctuations. This would give the market more time to absorb each burn event.

3. Implement a dynamic burn rate: Consider a dynamic burn rate mechanism which changes based on market conditions or token supply. This can be more adaptable and potentially prevent drastic price changes.

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.