



Building the Futuristic **Blockchain** Ecosystem

SECURITY AUDIT REPORT

SMARTAI

TABLE OF CONTENTS

02	Table of Contents	
03	Overview	
04	Contract Details	
05	Owner Privileges	
06	Audit Methodology	
07	Vulnerabilities Checklist	
08	Risk Classification	
09	Unit Tests	
10	Responsibility Disclaimer	
11	Inheritance Tree	
12	Function Details	
16	Manual Review	
17	Findings	
22	About Expelee	
23	Disclaimer	

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	Passed
Audit Date	28 April 2023

CONTRACT DETAILS

Token Name: SMARTAI

Symbol: SMARTAI

Network: Binance Smart Chain

Language: Solidity

Contract Address:

0x8E016C1F0d64956c302409F61c55532c8886F42C

Total Supply:5,000,000,000

Contract SHA-256 Checksum:

854a62fee8041932f1ab219ed9ab989b2813d96d

Owner's Wallet:

0x58922222bb0B8c7C17d8b30EAaC306d8b0fb1483

Deployer's Wallet:

0x58922222bb0B8c7C17d8b30EAaC306d8b0fb1483

OWNER PRIVILEGES

- Contract owner is able to set 20% for buy/sell fee separately (40% total)
- Contract owner is not able to set transfer tax (0% tax)
- Contract owner is not able to blacklist an arbitrary wallet
- Contract owner is not able to mint new tokens
- **Contract owner is able to disable buy/sell/transfers by setting a max transaction and max wallet amounts to a very little number (1 wei)**
- **Contract owner is able to set a limit for max amount of buy/holding/selling, lowest amount of this limit can be 0.000000000000000000000001 of SAT tokens (1 wei)**

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 Zeppelin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access 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 warnings that can remain unfixed.

Informational

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

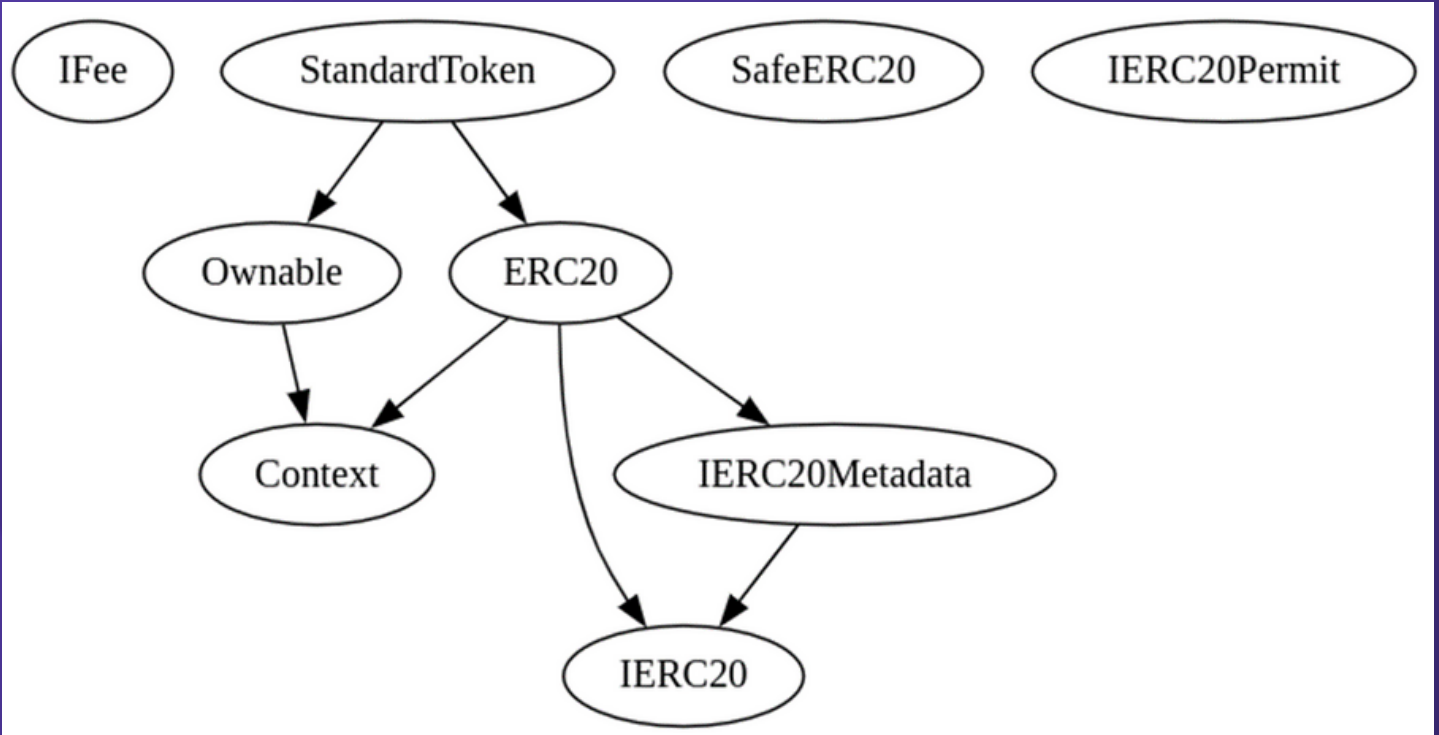
UNIT TESTS

In this security audit, our primary focus was on conducting a manual review of the SAT Token's smart contract codebase. Given that the token is already live, deployed on the Arbitrum blockchain, and actively being traded on the OeroSwap V2 platform, we deemed it sufficient to assess the contract's functioning through the available data on the block explorer. Consequently, we did not perform unit tests as part of this audit, and our analysis was limited to the manual code review and observations of the token's behavior in the live environment.



RESPONSIBILITY DISCLAIMER

It is important to note that this security audit was conducted after the SAT Token's ERC20 smart contract had already been deployed and started trading on the OeroSwap V2 platform. Our team bears no responsibility for any vulnerabilities, issues, or consequences that may have occurred or materialized prior to the commencement of this audit. The purpose of this audit is to assess the current state of the smart contract and provide recommendations to enhance its security and reliability. The SAT Token team is responsible for addressing any identified issues and ensuring the proper implementation of our recommendations.





INHERITANCE TREES



FUNCTION DETAILS

Symbol	Meaning
	Function can modify state
	Function is payable

```

| Contract |      Type      |      Bases      |      |      | |
|---|---|---|---|---|---|
| L | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| **IUniswapV2Router01** | Interface | |||
| L | factory | External | ! | | NO ! |
| L | WETH | External | ! | | NO ! |
| L | addLiquidity | External | ! | ● | NO ! |
| L | addLiquidityETH | External | ! |  | NO ! |
| L | removeLiquidity | External | ! | ● | NO ! |
| L | removeLiquidityETH | External | ! | ● | NO ! |
| L | removeLiquidityWithPermit | External | ! | ● | NO ! |
| L | removeLiquidityETHWithPermit | External | ! | ● | NO ! |
| L | swapExactTokensForTokens | External | ! | ● | NO ! |
| L | swapTokensForExactTokens | External | ! | ● | NO ! |
| L | swapExactETHForTokens | External | ! |  | NO ! |
| L | swapTokensForExactETH | External | ! | ● | NO ! |
| L | swapExactTokensForETH | External | ! | ● | NO ! |
| L | swapETHForExactTokens | External | ! |  | NO ! |
| L | quote | External | ! | | NO ! |
| L | getAmountOut | External | ! | | NO ! |
| L | getAmountIn | External | ! | | NO ! |
| L | getAmountsOut | External | ! | | NO ! |
| L | getAmountsIn | External | ! | | NO ! |
|||||
| **IUniswapV2Router02** | Interface | IUniswapV2Router01 |||
| L | removeLiquidityETHSupportingFeeOnTransferTokens | External | ! | ● | NO ! |
| L | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External | ! | ● | NO ! |
|
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External | ! | ● | NO ! |
| L | swapExactETHForTokensSupportingFeeOnTransferTokens | External | ! |  | NO ! |
| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External | ! | ● | NO ! |
|||||
| **IUniswapV2Pair** | Interface | |||
| L | name | External | ! | | NO ! |
| L | symbol | External | ! | | NO ! |
| L | decimals | External | ! | | NO ! |
| L | totalSupply | External | ! | | NO ! |
| L | balanceOf | External | ! | | NO ! |
| L | allowance | External | ! | | NO ! |
| L | approve | External | ! | ● | NO ! |
| L | transfer | External | ! | ● | NO ! |
| L | transferFrom | External | ! | ● | NO ! |
| L | DOMAIN_SEPARATOR | External | ! | | NO ! |
| L | PERMIT_TYPEHASH | External | ! | | NO ! |
| L | nonces | External | ! | | NO ! |
| L | permit | External | ! | ● | NO ! |

```

FUNCTION DETAILS

```

| L | MINIMUM_LIQUIDITY | External ! | | NO ! |
| L | factory | External ! | | NO ! |
| L | token0 | External ! | | NO ! |
| L | token1 | External ! | | NO ! |
| L | getReserves | External ! | | NO ! |
| L | price0CumulativeLast | External ! | | NO ! |
| L | price1CumulativeLast | External ! | | NO ! |
| L | kLast | External ! | | NO ! |
| L | mint | External ! | ● | NO ! |
| L | burn | External ! | ● | NO ! |
| L | swap | External ! | ● | NO ! |
| L | skim | External ! | ● | NO ! |
| L | sync | External ! | ● | NO ! |
| L | initialize | External ! | ● | NO ! |
|||||
| **IUniswapV2Factory** | Interface | |||
| L | feeTo | External ! | | NO ! |
| L | feeToSetter | External ! | | NO ! |
| L | getPair | External ! | | NO ! |
| L | allPairs | External ! | | NO ! |
| L | allPairsLength | External ! | | NO ! |
| L | createPair | External ! | ● | NO ! |
| L | setFeeTo | External ! | ● | NO ! |
| L | setFeeToSetter | External ! | ● | NO ! |
| L | INIT_CODE_PAIR_HASH | External ! | | NO ! |
|||||
| **IUniswapV2Caller** | Interface | |||
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ! | ● | NO ! |
|||||
| **IFee** | Interface | |||
| L | payFee | External ! | 🟢 | NO ! |
|||||
| **StandardToken** | Implementation | ERC20, Ownable |||
| L | <Constructor> | Public ! | 🟢 | ERC20 |
| L | decimals | Public ! | | NO ! |
| L | updateUniswapV2Pair | External ! | ● | onlyOwner |
| L | updateUniswapV2Router | Public ! | ● | onlyOwner |
| L | updateLiquidityFee | External ! | ● | onlyOwner |
| L | updateMaxWallet | External ! | ● | onlyOwner |
| L | updateMaxTransactionAmount | External ! | ● | onlyOwner |
| L | updateMarketingFee | External ! | ● | onlyOwner |
| L | updateMarketingWallet | External ! | ● | onlyOwner |
| L | updateMinAmountToTakeFee | External ! | ● | onlyOwner |
| L | setAutomatedMarketMakerPair | Public ! | ● | onlyOwner |
| L | _setAutomatedMarketMakerPair | Private 🔒 | ● | |
| L | excludeFromFee | External ! | ● | onlyOwner |

```


FUNCTION DETAILS

```

| L | excludeFromMaxTransactionAmount | External ! | ● | onlyOwner |
| L | _transfer | Internal 🔒 | ● | |
| L | takeFee | Private 🔒 | ● | lockTheSwap |
| L | swapTokensForBaseToken | Private 🔒 | ● | |
| L | addLiquidity | Private 🔒 | ● | |
| L | <Receive Ether> | External ! | 🟢 | NO ! |
|||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata |||
| L | <Constructor> | Public ! | ● | NO ! |
| L | name | Public ! | | NO ! |
| L | symbol | Public ! | | NO ! |
| L | decimals | Public ! | | NO ! |
| L | totalSupply | Public ! | | NO ! |
| L | balanceOf | Public ! | | NO ! |
| L | transfer | Public ! | ● | NO ! |
| L | allowance | Public ! | | NO ! |
| L | approve | Public ! | ● | NO ! |
| L | transferFrom | Public ! | ● | NO ! |
| L | increaseAllowance | Public ! | ● | NO ! |
| L | decreaseAllowance | Public ! | ● | NO ! |
| L | _transfer | Internal 🔒 | ● | |
| L | _mint | Internal 🔒 | ● | |
| L | _burn | Internal 🔒 | ● | |
| L | _approve | Internal 🔒 | ● | |
| L | _spendAllowance | Internal 🔒 | ● | |
| L | _beforeTokenTransfer | Internal 🔒 | ● | |
| L | _afterTokenTransfer | Internal 🔒 | ● | |
|||||
| **IERC20** | Interface | |||
| L | totalSupply | External ! | | NO ! |
| L | balanceOf | External ! | | NO ! |
| L | transfer | External ! | ● | NO ! |
| L | allowance | External ! | | NO ! |
| L | approve | External ! | ● | NO ! |
| L | transferFrom | External ! | ● | NO ! |
|||||
| **IERC20Metadata** | Interface | IERC20 |||
| L | name | External ! | | NO ! |
| L | symbol | External ! | | NO ! |
| L | decimals | External ! | | NO ! |
|||||
| **Context** | Implementation | |||
| L | _msgSender | Internal 🔒 | | |
| L | _msgData | Internal 🔒 | | |
|||||
| **Ownable** | Implementation | Context |||

```

FUNCTION DETAILS

```

| L | <Constructor> | Public ! | ● | NO ! |
| L | owner | Public ! | | NO ! |
| L | _checkOwner | Internal 🔒 | | |
| L | renounceOwnership | Public ! | ● | onlyOwner |
| L | transferOwnership | Public ! | ● | onlyOwner |
| L | _transferOwnership | Internal 🔒 | ● | |
|||||
| **SafeERC20** | Library | |||
| L | safeTransfer | Internal 🔒 | ● | |
| L | safeTransferFrom | Internal 🔒 | ● | |
| L | safeApprove | Internal 🔒 | ● | |
| L | safeIncreaseAllowance | Internal 🔒 | ● | |
| L | safeDecreaseAllowance | Internal 🔒 | ● | |
| L | safePermit | Internal 🔒 | ● | |
| L | _callOptionalReturn | Private 🔒 | ● | |
|||||
| **IERC20Permit** | Interface | |||
| L | permit | External ! | ● | NO ! |
| L | nonces | External ! | | NO ! |
| L | DOMAIN_SEPARATOR | External ! | | NO ! |
|||||
| **Address** | Library | |||
| L | isContract | Internal 🔒 | | |
| L | sendValue | Internal 🔒 | ● | |
| L | functionCall | Internal 🔒 | ● | |
| L | functionCall | Internal 🔒 | ● | |
| L | functionCallWithValue | Internal 🔒 | ● | |
| L | functionCallWithValue | Internal 🔒 | ● | |
| L | functionStaticCall | Internal 🔒 | | |
| L | functionStaticCall | Internal 🔒 | | |
| L | functionDelegateCall | Internal 🔒 | ● | |
| L | functionDelegateCall | Internal 🔒 | ● | |
| L | verifyCallResultFromTarget | Internal 🔒 | | |
| L | verifyCallResult | Internal 🔒 | | |
| L | _revert | Private 🔒 | | |

```

MANUAL REVIEW

Severity Criteria

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

Vulnerabilities are divided into three primary risk categories:

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			

FINDINGS

Findings	Severity	Found
High Risk	● High	2
Medium Risk	● Medium	0
Low Risk	● Low	0
Suggestion & discussion	● Informational	0
Gas Optimizations	● Gas Opt.	0

HIGH RISK FINDING

Max amount for buy/transfer/wallet/sells

Severity : High

Category: Centralization

Overview

The smart contract allows the owner to set maximum limits for buy, transfer, wallet amounts, and sell transactions. The owner can set these limits to as low as 1 wei (0.000000000000000001 SAT tokens), which could potentially disable buy, transfer, and sell operations for the SAT token. This high degree of control by the contract owner introduces centralization risks and could impact the token's overall functionality.

```
function updateMaxTransactionAmount(uint256
_maxTransactionAmount) external onlyOwner {
    require(_maxTransactionAmount > 0, "maxTransactionAmount > 0");
    emit UpdateMaxTransactionAmount(_maxTransactionAmount,
maxTransactionAmount);
    maxTransactionAmount = _maxTransactionAmount;
}
```

```
function updateMaxWallet(uint256 _maxWallet) external onlyOwner
{
    require(_maxWallet > 0, "maxWallet > 0");
    emit UpdateMaxWallet(_maxWallet, maxWallet);
    maxWallet = _maxWallet;
}
```

HIGH RISK FINDING

Suggestion :

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the maximum limits, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the maximum limits, adding an additional layer of security and reducing the centralization risk.

Issue Status: **Open**

HIGH RISK FINDING

Router can be upgraded

Severity : High

Category: Logical /Centralization

Overview

The smart contract allows the owner to update the mainRouter which is the contract that is used to add liquidity for the token during an internal swap. Setting mainRouter to a new contract that has some unknowns issues or doesn't support adding liquidity operations, can cause unknown issues for sells (potentially disabling them)

```
function updateUniswapV2Router(address newAddress) public
onlyOwner {
    require(
        newAddress != address(mainRouter),
        "The router already has that address"
    );
    emit UpdateUniswapV2Router(newAddress, address(mainRouter));
    mainRouter = IUniswapV2Router02(newAddress);
    address _mainPair =
    IUniswapV2Factory(mainRouter.factory()).createPair(
        address(this),
        baseTokenForPair
    );
    mainPair = _mainPair;
    _setAutomatedMarketMakerPair(mainPair, true);
}
```

HIGH RISK FINDING

Suggestion :

To mitigate this centralization issue, we propose the following options:

1. **Renounce Ownership:** Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. **Multi-signature Wallet:** Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.

Issue Status: **Open**

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.

The logo for Expelee, featuring the word "expelee" in a stylized font. The "ex" is in white, and "pelee" is in orange. The letters are bold and modern.

Building the Futuristic **Blockchain Ecosystem**