



SMART CONTRACT SECURITY AUDIT OF

Sapsali Coin



Summary - Preliminary Report

Audit Firm - Guardian Audits

Client Firm - Sapsali Token

Final Report Date - Preliminary Report

Audit Summary

After a line by line manual analysis and automated review, Guardian Audits has concluded that:

- Sapsali's smart contracts have a **HIGH RISK SEVERITY**
- Sapsali's smart contracts have an **ACTIVE OWNERSHIP**
- Important owner privileges – set fees, set max transaction amount, retain LP tokens, including/excluding addresses from rewards
- Sapsali's smart contract owner has multiple "write" privileges. Centralization risk correlated to the active ownership is **HIGH**

Notice that the examined smart contracts are not resistant to internal exploit. For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

 Blockchain network: **Fantom Opera**

 Verify the authenticity of this report on Guardian's GitHub: <https://github.com/guardianaudits>

Table of Contents

Project Information

Overview 4

Audit Scope & Methodology 5

Smart Contract Risk Assessment

Contract Overview 6

Inheritance Graph 10

Findings & Resolutions 11

Report Summary

Auditor’s Verdict 20

Addendum

Disclaimer 21

About Guardian Audits 22

Project Overview

Project Summary

Project Name	Sapsali Coin
Language	Solidity
Codebase	https://ftmscan.com/address/0xaf2ba90afe0cd84541cf32afb31fc86340a1d1b9#code
Commit	

Audit Summary

Delivery Date	Preliminary Report
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
● Critical	0	0	0	0	0	0
● High	3	3	0	0	0	0
● Medium	0	0	0	0	0	0
● Low	5	5	0	0	0	0

Audit Scope & Methodology

Scope

ID	File	SHA-1 Checksum
SAP	CoinToken.sol	494d2a494d049d19e37d2f99de74941e7a4d6506

Methodology

The auditing process pays special attention to the following considerations:



- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Vulnerability Classifications

Vulnerability Level	Classification
● Critical	Easily exploitable by anyone, causing loss/manipulation of assets or data.
● High	Arduously exploitable by a subset of addresses, causing loss/manipulation of assets or data.
● Medium	Inherent risk of future exploits that may or may not impact the smart contract execution.
● Low	Minor deviation from best practices

Contract Overview

Legend

	Symbol		Meaning	
	:-----:		-----	
			Function can modify state	
			Function is payable	

Contracts Description Table

	Contract		Type		Bases					
	:-----:		:-----:		:-----:		:-----:		:-----:	
	L		**Function Name**		**Visibility**		**Mutability**		**Modifiers**	
	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	
	SafeMath		Library							
	L add		Internal							
	L sub		Internal							
	L sub		Internal							
	L mul		Internal							
	L div		Internal							
	L div		Internal							
	L mod		Internal							
	L mod		Internal							
	Context		Implementation							
	L _msgSender		Internal							
	L _msgData		Internal							
	Address		Library							
	L isContract		Internal							
	L sendValue		Internal							
	L functionCall		Internal							
	L functionCall		Internal							
	L functionCallWithValue		Internal							
	L functionCallWithValue		Internal							
	L _functionCallWithValue		Private							

```

| **Ownable** | Implementation | Context |||
| L | owner | Public ! | |NO ! |
| L | renounceOwnership | Public ! | ● | onlyOwner |
| L | transferOwnership | Public ! | ● | onlyOwner |
| L | geUnlockTime | Public ! | |NO ! |
| L | lock | Public ! | ● | onlyOwner |
| L | unlock | Public ! | ● |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 ! |
| | | | |
| **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 ! |
| 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 ! |
| | | | |

```

```

| **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 ! |

```

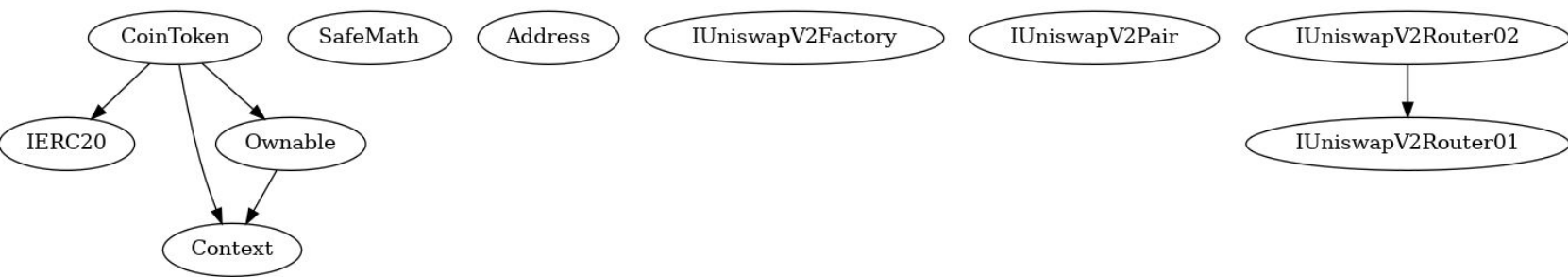


```

| **CoinToken** | Implementation | Context, IERC20, Ownable |||
| 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 | isExcludedFromReward | Public ! | | NO ! |
| L | totalFees | Public ! | | NO ! |
| L | deliver | Public ! | 🔴 | NO ! |
| L | reflectionFromToken | Public ! | | NO ! |
| L | tokenFromReflection | Public ! | | NO ! |
| L | excludeFromReward | Public ! | 🔴 | onlyOwner |
| L | includeInReward | External ! | 🔴 | onlyOwner |
| L | _transferBothExcluded | Private 🟡 | 🔴 | |
| L | excludeFromFee | Public ! | 🔴 | onlyOwner |
| L | includeInFee | Public ! | 🔴 | onlyOwner |
| L | setTaxFeePercent | External ! | 🔴 | onlyOwner |
| L | setLiquidityFeePercent | External ! | 🔴 | onlyOwner |
| L | setNumTokensSellToAddToLiquidity | Public ! | 🔴 | onlyOwner |
| L | setMaxTxPercent | Public ! | 🔴 | onlyOwner |
| L | setSwapAndLiquifyEnabled | Public ! | 🔴 | onlyOwner |
| L | <Receive Ether> | External ! | $🟢 | NO ! |
| L | _reflectFee | Private 🟡 | 🔴 | |
| L | _getValues | Private 🟡 | | |
| L | _getTValues | Private 🟡 | | |
| L | _getRValues | Private 🟡 | | |
| L | _getRate | Private 🟡 | | |
| L | _getCurrentSupply | Private 🟡 | | |
| L | _takeLiquidity | Private 🟡 | 🔴 | |
| L | claimTokens | Public ! | 🔴 | onlyOwner |
| L | calculateTaxFee | Private 🟡 | | |
| L | calculateLiquidityFee | Private 🟡 | | |
| L | removeAllFee | Private 🟡 | 🔴 | |
| L | restoreAllFee | Private 🟡 | 🔴 | |
| L | isExcludedFromFee | Public ! | | NO ! |
| L | _approve | Private 🟡 | 🔴 | |
| L | _transfer | Private 🟡 | 🔴 | |
| L | swapAndLiquify | Private 🟡 | 🔴 | lockTheSwap |
| L | swapTokensForEth | Private 🟡 | 🔴 | |
| L | addLiquidity | Private 🟡 | 🔴 | |
| L | _tokenTransfer | Private 🟡 | 🔴 | |
| L | _transferStandard | Private 🟡 | 🔴 | |
| L | _transferToExcluded | Private 🟡 | 🔴 | |
| L | _transferFromExcluded | Private 🟡 | 🔴 | |

```

Inheritance Graph



Findings & Resolutions

ID	Title	Category	Severity	Status
<u>SAP-1</u>	Centralization Risk	Centralization / Privilege	● High	Unresolved
<u>SAP-2</u>	Reentrancy Risk	Reentrancy Attack	● Low	Unresolved
<u>SAP-3</u>	Incorrect Error Message	Inaccuracy	● Low	Unresolved
<u>SAP-4</u>	Outdated method	Best Practices	● Low	Unresolved
<u>SAP-5</u>	DoS Risk	Denial-of-Service	● High	Unresolved
<u>SAP-6</u>	Balance Abuse	Centralization / Privilege	● High	Unresolved
<u>SAP-7</u>	Immutable Router	Centralization / Privilege	● Low	Unresolved
<u>SAP-8</u>	Misrepresented Ownership	Inaccuracy	● Low	Unresolved

SAP-1 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	● High	Global	Unresolved

Description

Contract does not have caps on token fees. Owner can call `setTaxFeePercent` and set the tax to an arbitrarily high amount leading to loss of user funds.

Similarly, the owner can call `setLiquidityFeePercent` and set it to an arbitrarily high amount.

Owner can call `setMaxTxPercent` and set it to 0. This would render `_transfer` useless alongside the methods that rely on it such as `addLiquidity` and `swapTokensForEth`

Owner address controls the liquidity added through `addLiquidityETH` (5th argument)

Recommendation

Set an immutable cap in the contract and add require statements in the functions responsible for setting fees so that the total cap cannot be exceeded.

Set owner to a multi-sig and introduce timelocking for changes.

Resolution

SAP-2 | Reentrancy Attack Risk

Category	Severity	Location	Status
Reentrancy	● Low	CoinToken.sol:1007-1051	Unresolved

Description

The `_transfer` function calls external contracts through `swapAndLiquify` and writes to state variables `_tOwned` and `_rOwned` afterwards. However, the router address is confirmed to be SpookySwap's and is immutable so there is not an imminent risk.

Recommendation

Implement the check-effects-interactions pattern or add a `nonReentrant` modifier through OpenZeppelin's `ReentrancyGuard.sol` trusted contract. Alternatively, extend the `inSwapAndLiquify` check to limit reentrancy in `_transfer` entirely

Resolution

SAP-3 | Incorrect Error Message

Category	Severity	Location	Status
Inaccuracy	● Low	CoinToken.sol:858	Unresolved

Description

Line 858 checks if the address is excluded to make it included. The message in the require should be “Account is already included”

Recommendation

Modify the message

Resolution

SAP-4 | Outdated Method

Category	Severity	Location	Status
Best Practices	● Low	CoinToken.sol:965	Unresolved

Description

Line 965 utilizes `.transfer()` which is limited to 2300 gas

Recommendation

Switch to `.call()` with a success check or use OpenZeppelin's `safeTransfer()`

Resolution

SAP-5 | Denial-of-Service Attack

Category	Severity	Location	Status
Best Practices	● High	CoinToken.sol:859,946	Unresolved

Description

The for loops in `_getCurrentSupply` and `includeInReward`, rely on the `_excluded` array which can vary in length. The owner can keep calling `excludeFromReward` and make the length of the `_excluded` array arbitrarily long which would lead to extreme gas costs that may exceed the block gas limit. This would render the token unusable as many functions, including `_transfer`, rely on `_getCurrentSupply`.

Recommendation

Reconsider if `excludeFromReward` is needed alongside `includeInReward`. If needed, switch to a multi-sig and timelocking setup.

Resolution

SAP-6 | Balance Abuse

Category	Severity	Location	Status
Best Practices	● High	CoinToken.sol:847-857	Unresolved

Description

The owner can call `excludeFromReward` on an address and include it later with `includeInReward` after rewards have accumulated. This results in the newly-included address gaining in balance while others lose some tokens due to the redistribution.

Recommendation

Reconsider if `excludeFromReward` is needed alongside `includeInReward`. If needed, switch to a multi-sig and timelocking setup.

Resolution

SAP-7 | Immutable Router Address

Category	Severity	Location	Status
Best Practices	<div><div></div>Low</div>	CoinToken.sol:705	Unresolved

Description

The contract does not allow for router upgrades and relies on the current SpookySwap router implementation.

Recommendation

Have a timelocked setter for the unirouter which allows you to keep the token logic same but decrease reliance on a third party.

Resolution

SAP-8 | Misrepresented ownership

Category	Severity	Location	Status
Inaccuracy	● Low	CoinToken.sol:457	Unresolved

Description

"Contract is locked until 7 days" is inaccurate as the lock time is arbitrarily set by the owner.

In addition, ownership can be renounced but then obtained again by unlocking which inaccurately represents "renouncing".

Recommendation

If possible, remove lock and unlock functionality as this is not the standard Ownable implementation. Otherwise, create a standard for the lock time.

Resolution

Auditor's Verdict

After a line by line manual analysis and automated review, Guardian Audits has concluded that:

- Sapsali's smart contracts have a **HIGH RISK SEVERITY**
- Sapsali's smart contracts have an **ACTIVE OWNERSHIP**
- Important owner privileges – set fees, set max transaction amount, retain LP tokens, including/excluding addresses from rewards
- Sapsali's smart contract owner has multiple "write" privileges. Centralization risk correlated to the active ownership is **HIGH**

Disclaimer

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Guardian to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Guardian’s position is that each company and individual are responsible for their own due diligence and continuous security. Guardian’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by Guardian is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract’s safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

About Guardian Audits

Founded in 2022 by DeFi experts, Guardian Audits is a leading audit firm in the DeFi smart contract space. With every audit report, Guardian Audits upholds best-in-class security while achieving our mission to relentlessly secure DeFi.

To learn more, visit <https://guardianaudits.com>

To view our audit portfolio, visit <https://github.com/guardianaudits>

To book an audit, message <https://t.me/guardianaudits>