Cyberscope

Audit Report

Simpsonsinu

May 2023

Table of Contents

Review

Contract Name	SIMPSONSINU
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Explorer	https://bscscan.com/address/0x5cc97dab7bc2c01556fbe3d06a09b8c559dff7d5
Address	0x5cc97dab7bc2c01556fbe3d06a09b8c559dff7d5
Network	BSC
Symbol	SIMPSONSINU
Decimals	9
Total Supply	420,690,000,000,000

Audit Updates

Initial Audit	16 May 2023

Source Files

Filename	SHA256
SIMPSONSINU.sol	0146256334b08a41b2fee6eeaedc155e647b0743d5cca4e65229d1b416248eb1

Findings Breakdown

	Critical	0
	Medium	1
	Minor / Informative	6

Severity	Unresolved	Acknowledged	Resolved	Other
● Critical	0	0	0	0
● Medium	1	0	0	0
● Minor / Informative	6	0	0	0

Analysis

● Critical   ● Medium   ● Minor / Informative   ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

🔴 Critical		🟠 Medium		⚪ Minor / Informative

Severity	Code	Description	Status
⚪	RED	Redundant Event Declaration	Unresolved
🟠	CSD	Circulating Supply Discrepancy	Unresolved
⚪	RS	Redundant Struct	Unresolved
⚪	IDI	Immutable Declaration Improvement	Unresolved
⚪	L02	State Variables could be Declared Constant	Unresolved
⚪	L04	Conformance to Solidity Naming Conventions	Unresolved
⚪	L14	Uninitialized Variables in Local Scope	Unresolved

RED - Redundant Event Declaration

Criticality	Minor / Informative
Location	SIMPSONSINU.sol#L320,322
Status	Unresolved

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The contract declares the events `SwapAndLiquify` and `SwapTokensAtAmountUpdated` but they are not used in the contract. As a result, both events are redundant.

```solidity
event SwapAndLiquify(uint256 tokensSwapped, uint256 bnbReceived, uint256
tokensIntoLiqudity);
event SwapTokensAtAmountUpdated(uint256 amount);
```

Recommendation

The team is advised to take these segments into consideration and rewrite them so the runtime will be more performant. That way it will improve the efficiency and performance of the source code and reduce the cost of executing it.

CSD - Circulating Supply Discrepancy

Criticality	Medium
Location	SIMPSONSINU.sol#L385
Status	Unresolved

Description

According to the ERC20 specification, the `totalSupply()` function should return the total supply of the token. The total supply should always equal the sum of the balances. The contract does not return the `totalSupply()`. Instead, the function returns the `totalSupply()` minus the amount that has been moved to the dead address. This amount is the circulating supply of the token. Many decentralized applications and tools are calculating many indicators like the circulating supply and market cap based on the `totalSupply()`. As a result, these applications will produce misleading results.

```solidity
function totalSupply() public view override returns (uint256) {
    return _tTotalVisual;
}
```

Recommendation

The `totalSupply()` should always equal the sum of the holder's balances. The contract should comply with this convention so that the decentralized applications will produce correct results.

RS - Redundant Struct

Criticality	Minor / Informative
Location	SIMPSONSINU.sol#L304
Status	Unresolved

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The contract declares the `Taxes` struct along with the `taxes` variable. This variable is not being used in the contract. As a result, it is redundant.

```solidity
struct Taxes {
    uint256 reflectionFee;
    uint256 marketingFee;
    uint256 buyBackFee;
}

Taxes public taxes = Taxes(3, 5, 1);
```

Recommendation

The team is advised to take these segments into consideration and rewrite them so the runtime will be more performant. That way it will improve the efficiency and performance of the source code and reduce the cost of executing it.

IDI - Immutable Declaration Improvement

Criticality	Minor / Informative
Location	SIMPSONSINU.sol#L355
Status	Unresolved

Description

The contract is using variables that initialize them only in the constructor. The other functions are not mutating the variables. These variables are not defined as `immutable`.

```
swapTokensAtAmount
```

Recommendation

By declaring a variable as immutable, the Solidity compiler is able to make certain optimizations. This can reduce the amount of storage and computation required by the contract, and make it more gas-efficient.

L02 - State Variables could be Declared Constant

Criticality	Minor / Informative
Location	SIMPSONSINU.sol#L279,280,281
Status	Unresolved

Description

State variables can be declared as constant using the constant keyword. This means that the value of the state variable cannot be changed after it has been set. Additionally, the constant variables decrease gas consumption of the corresponding transaction.

```
string private _name     = "The Simpsons Inu"
string private _symbol   = "SIMPSONSINU"
uint8  private _decimals = 9
```

Recommendation

Constant state variables can be useful when the contract wants to ensure that the value of a state variable cannot be changed by any function in the contract. This can be useful for storing values that are important to the contract's behavior, such as the contract's address or the maximum number of times a certain function can be called. The team is advised to add the constant keyword to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	SIMPSONSINU.sol#L98,99,115,134,285,525,529,533
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
function DOMAIN_SEPARATOR() external view returns (bytes32);
function PERMIT_TYPEHASH() external pure returns (bytes32);
function MINIMUM_LIQUIDITY() external pure returns (uint);
function WETH() external pure returns (address);
uint256 internal _tTotalVisual = 420_690_000_000_000 * (10 ** _decimals)
uint256 _amount
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.

Find more information on the Solidity documentation

https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L14 - Uninitialized Variables in Local Scope

Criticality	Minor / Informative
Location	SIMPSONSINU.sol#L326
Status	Unresolved

Description

Using an uninitialized local variable can lead to unpredictable behavior and potentially cause errors in the contract. It's important to always initialize local variables with appropriate values before using them.

```
address router
```

Recommendation

By initializing local variables before using them, the contract ensures that the functions behave as expected and avoid potential issues.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Address	Library			

	sendValue	Internal	✓	
IUniswapV2Fac tory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-

	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-

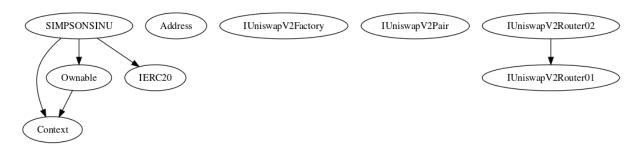|				
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2 Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-

SIMPSONSINU	Implementation	Context, IERC20, Ownable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	isExcludedFromReward	Public		-
	totalReflectionDistributed	Public		-
	reflectionFromToken	Public		-
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
		External	Payable	-
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
SIMPSONSINU				

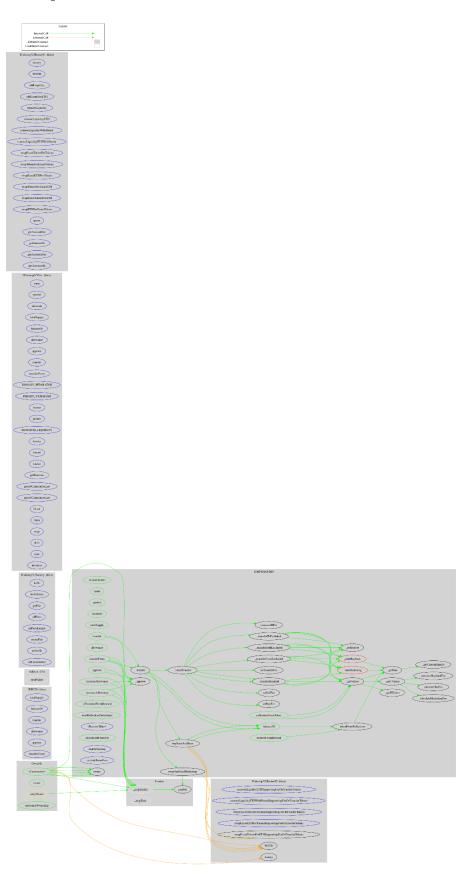|				
	_getRate	Private		
	_getCurrentSupply	Private		
	_takeBuyback	Private	✓	
	_takeMarketing	Private	✓	
	calculateTaxFee	Private		
	calculateBuybackFee	Private		
	calculateMarketingFee	Private		
	removeAllFee	Private	✓	
	setBuyFee	Private	✓	
	setSellFee	Private	✓	
	setTransferFee	Private	✓	
	isExcludedFromFee	Public		-
	_approve	Private	✓	
	enableTrading	External	✓	onlyOwner
	_transfer	Private	✓	
	buyBackAndBurn	Private	✓	
	swapAndSendMarketing	Private	✓	
	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	
	_transferBothExcluded	Private	✓	
	excludeFromFees	External	✓	onlyOwner

Inheritance Graph

Flow Graph

Summary

Simpsonsinu contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. Simpsonsinu is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler errors or critical issues. The contract owner has renounced ownership. There is also a fixed fee of 9%.

Renounce Ownership Transaction

https://bscscan.com/tx/0x3e0a13e2305a7efacba22f3fc8703e0d33ef144596b8bdf4cabea24409b83839

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. 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 regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document 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 represents an extensive assessment 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 Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope'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 Cyberscope are subject to dependencies and are 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.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io