



Cyberscope

Audit Report

SQUT Meme Token

August 2024

Network BSC

Address 0xA5E94E1CD621c0F6361C57c39712d777ebcd4146

Audited by © cyberscope

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	CR	Code Repetition	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L16	Validate Variable Setters	Unresolved

Table of Contents

Analysis	1
Diagnostics	2
Table of Contents	3
Risk Classification	4
Review	5
Audit Updates	5
Source Files	5
Findings Breakdown	6
CR - Code Repetition	7
Description	7
Recommendation	8
L04 - Conformance to Solidity Naming Conventions	9
Description	9
Recommendation	9
L16 - Validate Variable Setters	10
Description	10
Recommendation	10
Functions Analysis	11
Inheritance Graph	13
Flow Graph	14
Summary	15
Disclaimer	16
About Cyberscope	17

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation:** This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation:** This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical:** Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium:** Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor:** Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative:** Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
● Critical	Highly Likely / High Impact
● Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
● Minor / Informative	Unlikely / Low to no Impact

Review

Contract Name	DefiToken
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	1337 runs
Explorer	https://bscscan.com/address/0xa5e94e1cd621c0f6361c57c39712d777ebcd4146
Address	0xa5e94e1cd621c0f6361c57c39712d777ebcd4146
Network	BSC
Symbol	SQUT
Decimals	16
Total Supply	254,999,998,999,990
Badge Eligibility	Yes

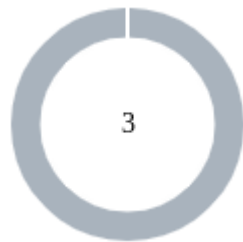
Audit Updates

Initial Audit	26 Aug 2024
---------------	-------------

Source Files

Filename	SHA256
contracts/DefiToken.sol	1860c44acaef7dcdb015beec9dde10c601b31cc8706af1e6fe148b0857ebdb5
contracts/lib/LibCommon.sol	ad40e79524942f0927be19739e7c96b7a52147f5cf54fdd7eb676720db70b66a

Findings Breakdown



● Critical	0
● Medium	0
● Minor / Informative	3

Severity	Unresolved	Acknowledged	Resolved	Other
● Critical	0	0	0	0
● Medium	0	0	0	0
● Minor / Informative	3	0	0	0

CR - Code Repetition

Criticality	Minor / Informative
Location	contracts/DefiToken.sol#L239,268
Status	Unresolved

Description

The contract contains repetitive code segments. There are potential issues that can arise when using code segments in Solidity. Some of them can lead to issues like gas efficiency, complexity, readability, security, and maintainability of the source code. It is generally a good idea to try to minimize code repetition where possible.

Specifically the `transfer` and `transferFrom` functions share similar code segments.


```
function transfer(
    address to,
    uint256 amount
) public virtual override returns (bool) {
    uint256 taxAmount = _taxAmount(msg.sender, amount);
    uint256 deflationAmount = _deflationAmount(amount);
    uint256 amountToTransfer = amount - taxAmount -
deflationAmount;

    if (isMaxAmountOfTokensSet()) {
        if (balanceOf(to) + amountToTransfer >
maxTokenAmountPerAddress) {
            revert DestBalanceExceedsMaxAllowed(to);
        }
    }

    if (taxAmount != 0) {
        _transfer(msg.sender, taxAddress, taxAmount);
    }
    if (deflationAmount != 0) {
        _burn(msg.sender, deflationAmount);
    }
    return super.transfer(to, amountToTransfer);
}

function transferFrom(
    address from,
    address to,
    uint256 amount
) public virtual override returns (bool) {
    ...
    return super.transferFrom(from, to, amountToTransfer);
}
```

Recommendation

The team is advised to avoid repeating the same code in multiple places, which can make the contract easier to read and maintain. The authors could try to reuse code wherever possible, as this can help reduce the complexity and size of the contract. For instance, the contract could reuse the common code segments in an internal function in order to avoid repeating the same code in multiple places.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/DefiToken.sol#L205,206,223
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.

```
address _taxAddress
uint256 _taxBPS
uint256 _deflationBPS
```

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/stable/style-guide.html#naming-conventions>.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	contracts/DefiToken.sol#L99,113,215
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
taxAddress = _taxAddress  
initialTokenOwner = tokenOwner
```

Recommendation

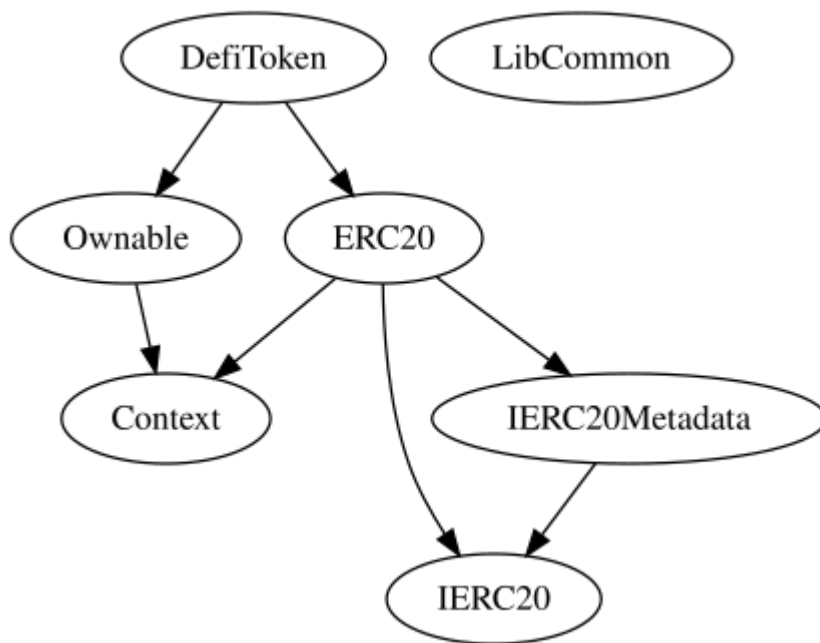
By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
DefiToken	Implementation	ERC20, Ownable		
		Public	✓	ERC20
	isMintable	Public		-
	isBurnable	Public		-
	isMaxAmountOfTokensSet	Public		-
	isDocumentUriAllowed	Public		-
	decimals	Public		-
	isTaxable	Public		-
	isDeflationary	Public		-
	setDocumentUri	External	✓	onlyOwner
	setMaxTokenAmountPerAddress	External	✓	onlyOwner
	setTaxConfig	External	✓	onlyOwner
	setDeflationConfig	External	✓	onlyOwner
	transfer	Public	✓	-
	transferFrom	Public	✓	-
	mint	External	✓	onlyOwner
	burn	External	✓	onlyOwner
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_taxAmount	Internal		

	_deflationAmount	Internal		
--	------------------	----------	--	--

Inheritance Graph



Flow Graph



Summary

SQUT meme token contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. SQUT meme token is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

The contract's ownership has been renounced. The information regarding the transaction can be accessed through the following link:

<https://bscscan.com/tx/0x5570b69e4cceaf735ca1d38936ff3b9ffc9910bc54cec83e1bb9220bd1389547>

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

cyberscope.io