Cyberscope

Audit Report

SQUT Meme Token

April 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	Unresolved
●	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

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	499,999,998,999,990
Badge Eligibility	Yes

Audit Updates

Initial Audit	07 Apr 2024

Source Files

Filename	SHA256
contracts/DefiToken.sol	1860c44acaef7dcdb015beec9dde10c601b31cc8706af1e6fe148b0857ebbdb5
contracts/lib/LibCommon.sol	ad40e79524942f0927be19739e7c96b7a52147f5cf54fdd7eb676720db70b66a

@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/token/ERC20/IERC20.sol	7ebde70853ccafcf1876900dad458f46eb9444d591d39bfc58e952e2582f5587
@openzeppelin/contracts/token/ERC20/ERC20.sol	d20d52b4be98738b8aa52b5bb0f88943f62128969b33d654fbca731539a7fe0a
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	af5c8a77965cc82c33b7ff844deb9826166689e55dc037a7f2f790d057811990
@openzeppelin/contracts/access/Ownable.sol	a8e4e1ae19d9bd3e8b0a6d46577eec098c01fbaffd3ec1252fd20d799e73393b

Findings Breakdown

Critical	1
Medium	0
Minor / Informative	3

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	0	0	0	0
Minor / Informative	3	0	0	0

ELFM - Exceeds Fees Limit

Criticality	Critical
Location	contracts/DefiToken.sol#L204
Status	Unresolved

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setTaxConfig` function with a high percentage value.

```
function setTaxConfig(
    address _taxAddress,
    uint256 _taxBPS
) external onlyOwner {
    if (!isTaxable()) {
        revert TokenIsNotTaxable();
    }
    if (_taxBPS > MAX_ALLOWED_BPS) {
        revert InvalidTaxBPS(_taxBPS);
    }
    LibCommon.validateAddress(_taxAddress);
    taxAddress = _taxAddress;
    taxBPS = _taxBPS;
    emit TaxConfigSet(_taxAddress, _taxBPS);
}
```

Recommendation

The contract could embody a check for the maximum acceptable value. The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

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/v0.8.17/style-guide.html#naming-convention.

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

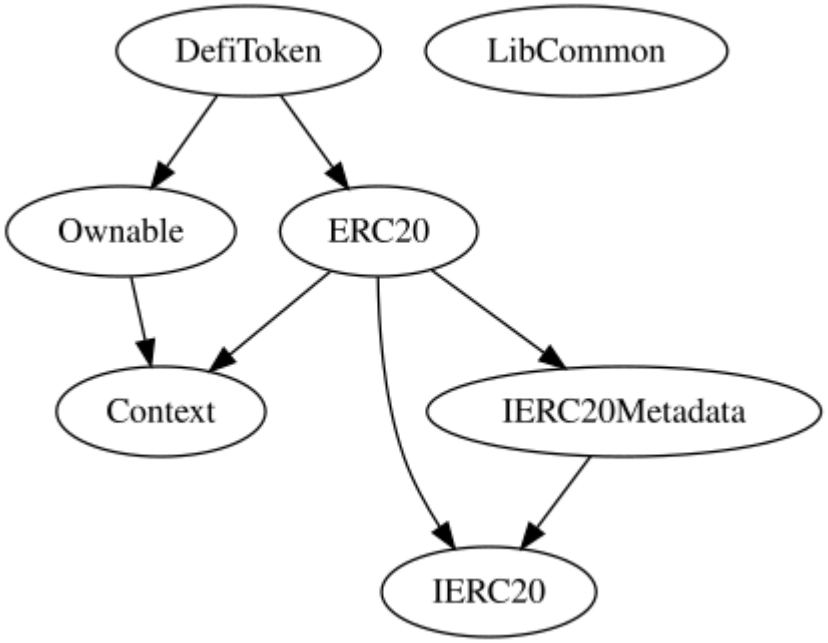
By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

Functions Analysis

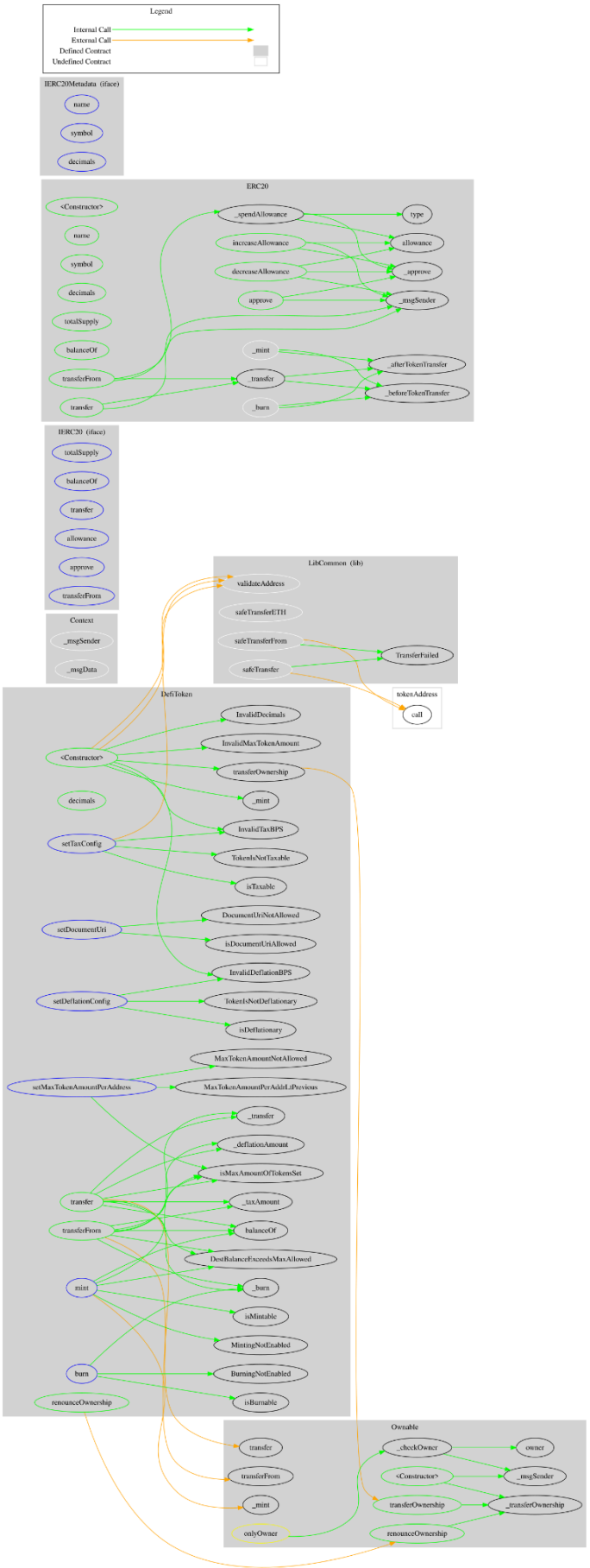
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
DefiToken	Implementation	ERC20, Ownable		
		Public	✓	ERC20
	isMintable	Public		-
	isBurnable	Public		-
	isMaxAmountOfTokensSet	Public		-
	isDocumentUriAllowed	Public		-
	decimals	Public		-
	isTaxable	Public		-
	isDeflationary	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


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IERC20Metadata (iface)
name
symbol
decimals
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
transferFrom
transfer
_spendAllowance
increaseAllowance
decreaseAllowance
approve
type
allowance
_approve
_msgSender
_mint
_transfer
_burn
_afterTokenTransfer
_beforeTokenTransfer
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
_msgData
LibCommon (lib)
validateAddress
safeTransferETH
safeTransferFrom
safeTransfer
TransferFailed
tokenAddress
call
DefiToken
<Constructor>
decimals
setTaxConfig
setDocumentUri
setDeflationConfig
setMaxTokenAmountPerAddress
transfer
transferFrom
mint
burn
renounceOwnership
InvalidDecimals
InvalidMaxTokenAmount
transferOwnership
_mint
InvalidTaxBPS
TokenIsNotTaxable
isTaxable
DocumentUriNotAllowed
isDocumentUriAllowed
InvalidDeflationBPS
TokenIsNotDeflationary
isDeflationary
MaxTokenAmountNotAllowed
MaxTokenAmountPerAddrLtPrevious
_transfer
_deflationAmount
isMaxAmountOfTokensSet
_taxAmount
balanceOf
DestBalanceExceedsMaxAllowed
_burn
isMintable
MintingNotEnabled
BurningNotEnabled
isBurnable
Ownable
transfer
transferFrom
_mint
onlyOwner
_checkOwner
<Constructor>
transferOwnership
renounceOwnership
owner
_msgSender
_transferOwnership

Summary

SQUT meme token contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like manipulate the fees. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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