



QuillAudits



Audit Report  
August, 2021





# Contents

Scope of Audit	01
Techniques and Methods	02
Issue Categories	03
Issues Found – Code Review/Manual Testing	04
Automated Testing	09
Disclaimer	13
Summary	14

## Scope of Audit

The scope of this audit was to analyze and document the EXIPTOKEN smart contract codebase for quality, security, and correctness.

## Checked Vulnerabilities

We have scanned the smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level



## Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

### Structural Analysis

In this step we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

SmartCheck.

### Static Analysis

Static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step a series of automated tools are used to test security of smart contracts.

### Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

### Gas Consumption

In this step we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and possibilities of optimization of code to reduce gas consumption.



## Tools and Platforms used for Audit

Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

## Issue Categories

Every issue in this report has been assigned with a severity level. There are four levels of severity and each of them has been explained below.

### High severity issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality and we recommend these issues to be fixed before moving to a live environment.

### Medium level severity issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems and they should still be fixed.

### Low level severity issues

Low level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

### Informational

These are severity four issues which indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

## Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	0	2
Acknowledged	0	0	0	0
Closed	0	0	0	0

## Introduction

During the period of **August 9,2021 to August 11, 2021** - QuillAudits Team performed a security audit for EXIPTOKEN smart contracts.

The code for the audit was taken from following the official link:

File: EXIPTOKEN.sol

Smart Contract Online Code:

<https://bscscan.com/address/0xb17a39f8177894aa66a706cafecb85ee3ecf504d#code>

Note	Date	MD5 hash
Version 1	10/08/2021	<u>F82E98C092E03290A77066CD77C7A527</u>



# Issues Found – Code Review / Manual Testing

## High severity issues

No issues were found.

## Medium severity issues

No issues were found.

## Low level severity issues

No issues were found.

## Informational

### 1. Use the latest solidity version

```
pragma solidity ^0.8.0;
```

#### Description

Using the latest solidity will prevent any compiler-level bugs.

#### Remediation

Please use 0.8.6 which is the latest version.

Status: Open

### 2. Missing error message

Line	Code
889	<pre>function pause() public {     require(hasRole(DEFAULT_ADMIN_ROLE, msg.sender));     _pause(); }</pre>
894	<pre>function unpause() public {     require(hasRole(DEFAULT_ADMIN_ROLE, msg.sender));     _unpause(); }</pre>

### Description

There are required conditions but not added error messages.

### Remediation

Put a relevant error message.

**Status:** Open



# Functional test

Function Names	Testing results
pause	Missing error message
unpause	Missing error message
_beforeTokenTransfer	Passed
name	Passed
symbol	Passed
decimals	Passed
totalSupply	Passed
balanceOf	Passed
transfer	Passed
allowance	Passed
approve	Passed
transferFrom	Passed
increaseAllowance	Passed
decreaseAllowance	Passed
_transfer	Passed
_mint	Passed
_burn	Passed
_approve	Passed
_beforeTokenTransfer	Passed
onlyRole	Passed



Function Names	Testing results
supportsInterface	Passed
hasRole	Passed
_checkRole	Passed
getRoleAdmin	Passed
grantRole	Passed
revokeRole	Passed
renounceRole	Passed
_setupRole	Passed
_setRoleAdmin	Passed
_grantRole	Passed
_revokeRole	Passed
paused	Passed
whenNotPaused	Passed
whenPaused	Passed
_pause	Passed
_unpause	Passed



# Automated Testing

## Slither

```
INFO:Detectors:
AccessControl.setRoleAdmin(bytes32,bytes32) (EXIPTOKEN.sol#751-754) is never used and should be removed
Context._msgData() (EXIPTOKEN.sol#118-121) is never used and should be removed
ERC20._burn(address,uint256) (EXIPTOKEN.sol#371-382) is never used and should be removed
Strings.toHexString(uint256) (EXIPTOKEN.sol#457-468) is never used and should be removed
Strings.toString(uint256) (EXIPTOKEN.sol#432-452) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (EXIPTOKEN.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (EXIPTOKEN.sol#534) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Constant Strings.alphabet (EXIPTOKEN.sol#427) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (EXIPTOKEN.sol#119)" inContext (EXIPTOKEN.sol#113-122)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
EXIPTOKEN.constructor(address) (EXIPTOKEN.sol#859-862) uses literals with too many digits:
- _mint(admin,2100000 * 10 ** decimals()) (EXIPTOKEN.sol#861)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
name() should be declared external:
- ERC20.name() (EXIPTOKEN.sol#176-178)
symbol() should be declared external:
- ERC20.symbol() (EXIPTOKEN.sol#184-186)
totalSupply() should be declared external:
- ERC20.totalSupply() (EXIPTOKEN.sol#208-210)
balanceOf(address) should be declared external:
- ERC20.balanceOf(address) (EXIPTOKEN.sol#215-217)
transfer(address,uint256) should be declared external:
- ERC20.transfer(address,uint256) (EXIPTOKEN.sol#227-230)
allowance(address,address) should be declared external:
- ERC20.allowance(address,address) (EXIPTOKEN.sol#235-237)
approve(address,uint256) should be declared external:
- ERC20.approve(address,uint256) (EXIPTOKEN.sol#246-249)

transferFrom(address,address,uint256) should be declared external:
- ERC20.transferFrom(address,address,uint256) (EXIPTOKEN.sol#264-272)
increaseAllowance(address,uint256) should be declared external:
- ERC20.increaseAllowance(address,uint256) (EXIPTOKEN.sol#286-289)
decreaseAllowance(address,uint256) should be declared external:
- ERC20.decreaseAllowance(address,uint256) (EXIPTOKEN.sol#305-311)
grantRole(bytes32,address) should be declared external:
- AccessControl.grantRole(bytes32,address) (EXIPTOKEN.sol#689-691)
revokeRole(bytes32,address) should be declared external:
- AccessControl.revokeRole(bytes32,address) (EXIPTOKEN.sol#702-704)
renounceRole(bytes32,address) should be declared external:
- AccessControl.renounceRole(bytes32,address) (EXIPTOKEN.sol#720-724)
pause() should be declared external:
- EXIPTOKEN.pause() (EXIPTOKEN.sol#864-867)
unpause() should be declared external:
- EXIPTOKEN.unpause() (EXIPTOKEN.sol#869-872)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:EXIPTOKEN.sol analyzed (11 contracts with 75 detectors), 26 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

## Results

No major issues were found. Some false positive errors were reported by the tool. All the other issues have been categorized above according to their level of severity.



# SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

contracts/EXIPTOKEN.sol

Security

Transaction origin:

INTERNAL ERROR in module Transaction origin: can't convert undefined to object

Pos: not available

Check-effects-interaction:

INTERNAL ERROR in module Check-effects-interaction: can't convert undefined to object

Pos: not available

Inline assembly:

INTERNAL ERROR in module Inline assembly: can't convert undefined to object

Pos: not available

Block timestamp:

INTERNAL ERROR in module Block timestamp: can't convert undefined to object

Pos: not available

Low level calls:

INTERNAL ERROR in module Low level calls: can't convert undefined to object

Pos: not available

Selfdestruct:

INTERNAL ERROR in module Selfdestruct: can't convert undefined to object

Pos: not available

Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object

Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object

Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object

Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object

Pos: not available



## ERC

### ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object  
Pos: not available

## Miscellaneous

### Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object  
Pos: not available

### Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object  
Pos: not available

### No return:

INTERNAL ERROR in module No return: can't convert undefined to object  
Pos: not available

### Guard conditions:

INTERNAL ERROR in module Guard conditions: can't convert undefined to object  
Pos: not available

### String length:

INTERNAL ERROR in module String length: can't convert undefined to object  
Pos: not available

## SOLHINT LINTER

contracts/EXIPTOKEN.sol:8:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:85:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:110:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:134:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:178:5: Error: Explicitly mark visibility in function (Set ignoreConstructors to true if using solidity >=0.7.0)

contracts/EXIPTOKEN.sol:429:94: Error: Code contains empty blocks

contracts/EXIPTOKEN.sol:434:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:439:30: Error: Constant name must be in capitalized SNAKE\_CASE

contracts/EXIPTOKEN.sol:501:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:525:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:551:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:790:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:816:5: Error: Explicitly mark visibility in function (Set ignoreConstructors to true if using solidity >=0.7.0)

contracts/EXIPTOKEN.sol:878:1: Error: Compiler version ^0.8.0 does not satisfy the r semver requirement

contracts/EXIPTOKEN.sol:884:5: Error: Explicitly mark visibility in function (Set ignoreConstructors to true if using solidity >=0.7.0)



## Disclaimer

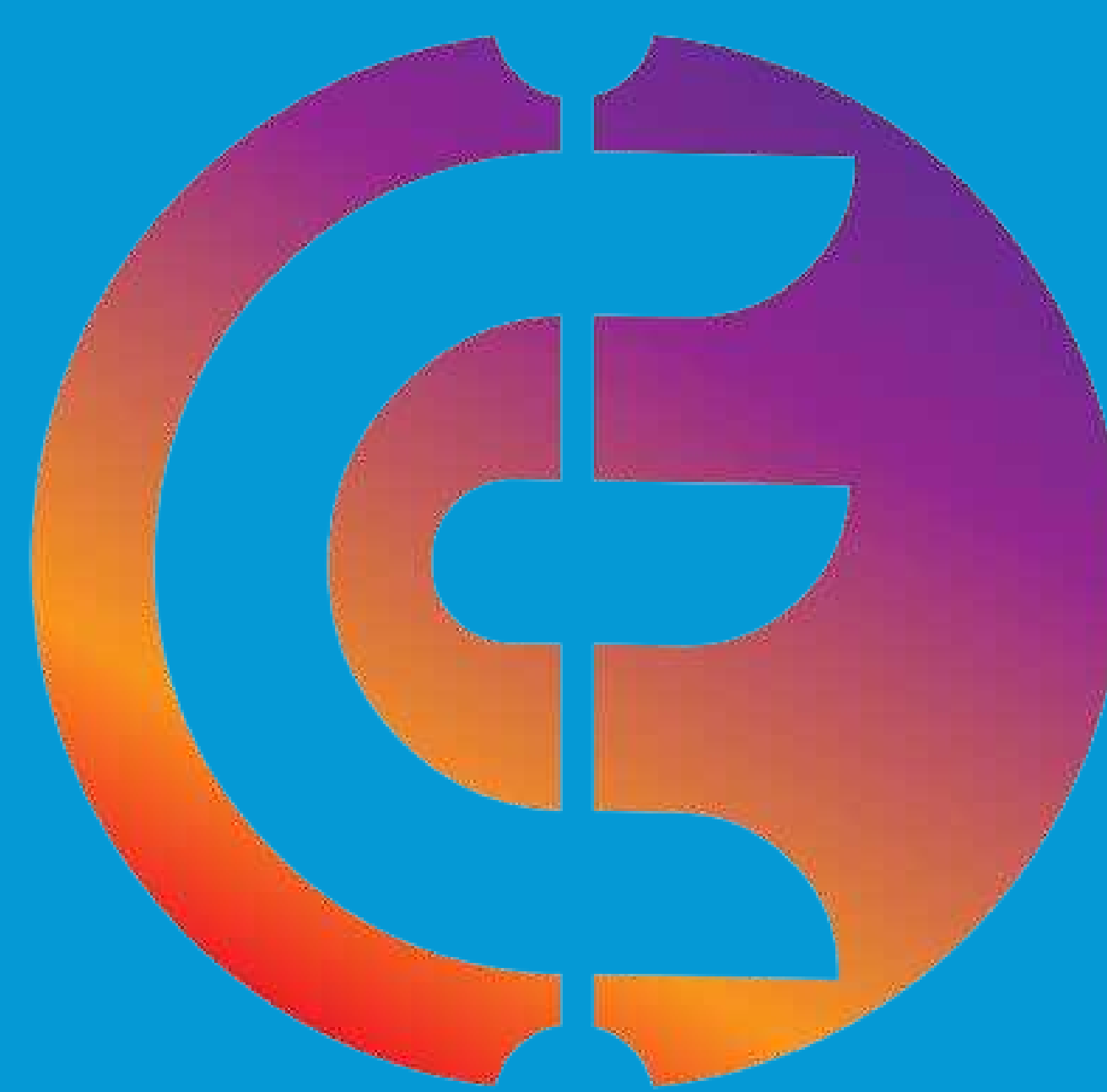
Quillhash audit is not a security warranty, investment advice, or an endorsement of the EXIPTOKEN platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the EXIPTOKEN Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

## Closing Summary

Overall, smart contracts are very well written and adhere to guidelines.

No instances of Integer Overflow and Underflow vulnerabilities or Back-Door Entry were found in the contract, but relying on other contracts might cause Reentrancy Vulnerability.





**QuillAudits**



Canada, India, Singapore and United Kingdom



[audits.quillhash.com](https://audits.quillhash.com)



[audits@quillhash.com](mailto:audits@quillhash.com)