QuillAudits

Audit Report
May, 2021

Contents

Scope of Audit	01
Techniques and Methods	01
Issue Categories	02
Introduction	04
Issues Found – Code Review/Manual Testing	04
Summary	07
Disclaimer	08

Scope of Audit

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

Check Vulnerabilities

- 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 contracts 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

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

Introduction

During the period of April 19th, 2021 to April 21st, 2021 - Quillhash Team performed a security audit for TycoonToken smart contracts.

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

https://etherscan.io/address/0x3A82D3111aB5faF39d847D46023d9090261A658F#code

Issues Found – Code Review / Manual Testing

A. Contract Name - TycoonToken

High Severity Issues

Not Found

Medium severity issues

Not Found

Low level severity issues

1. **User ETHER Units instead of Large Digits**
   Line no: 24-37

   Description:
   Since the decimal points used for the Tycoon token is **18**, the globally available Ether Units can be used instead of multiplying token allocation amounts with 10^18 while assigning them to State Variables. This will enhance the readability of the contract code.

   Solidity provides some globally available units like **ether** which symbolizes **10^18**.
   For instance,
   “uint256 private constant SALE_TOTAL = 84000000*1018”** can simply be written as
   “uint256 private constant SALE_TOTAL = 84000000 ether”

Recommendation:
Consider using **Ether Units(ether)** instead of multiplying token allocation amounts with **(10**18).**

2. **Redundant Require Statement Check**
   Line no: 783

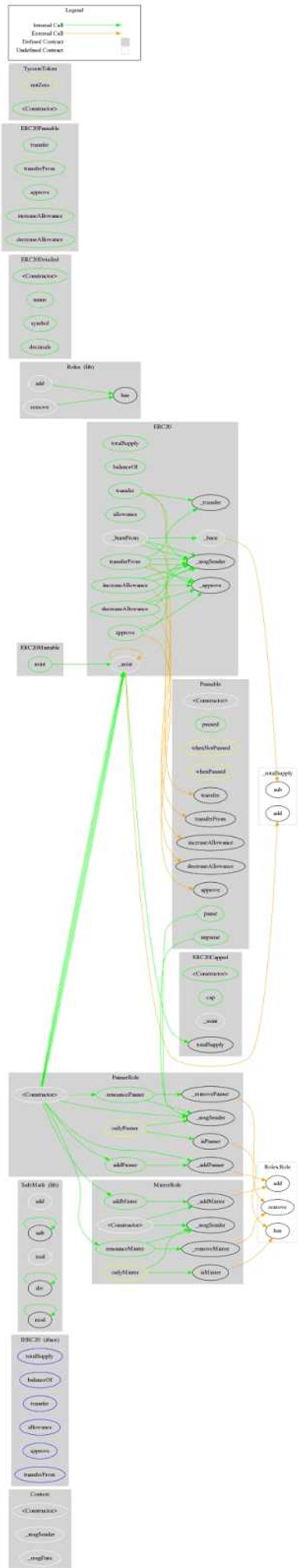
   Description:
   The constructor of the **TycoonToken** contract includes a **require** statement at Line 783 that validates that the total token allocations are as expected.

   Such validations are more effective when the token allocation amounts are passed through constructor arguments to ensure that no wrong value is passed.

   However, since the token allocation amounts are already initialized and hard-coded in the contract's State variables itself, the **require** statement does not hold much significance and affects the gas optimization of the contract.

   Recommendation:
   Redundant use of require statements should be avoided.

   Contract Control Flow



Legend
Internal Call
External Call
Defined Contract
Undefined Contract
ERC20
totalSupply
balanceOf
transfer
allowance
transferFrom
increaseAllowance
decreaseAllowance
approve
_mint
_msgSender
_approve
ERC20Mintable
Pausable
<Constructor>
pause
unpause
ERC20Capped
cap
PauserRole
onlyPauser
isPauser
addPauser
MinterRole
addMinter
renounceMinter
onlyMinter
isMinter
Roles.Role
add
sub
has
remove
Context
_msgData

Closing Summary

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

During the process of audit, No issues of high, medium severity were found. However, some low severity issues were found and have been documented above.

Moreover, No instances of Re-entrancy or Back-Door Entry were found in the contract.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the **TycoonToken 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 **TycoonToken** Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

hello@quillhash.com