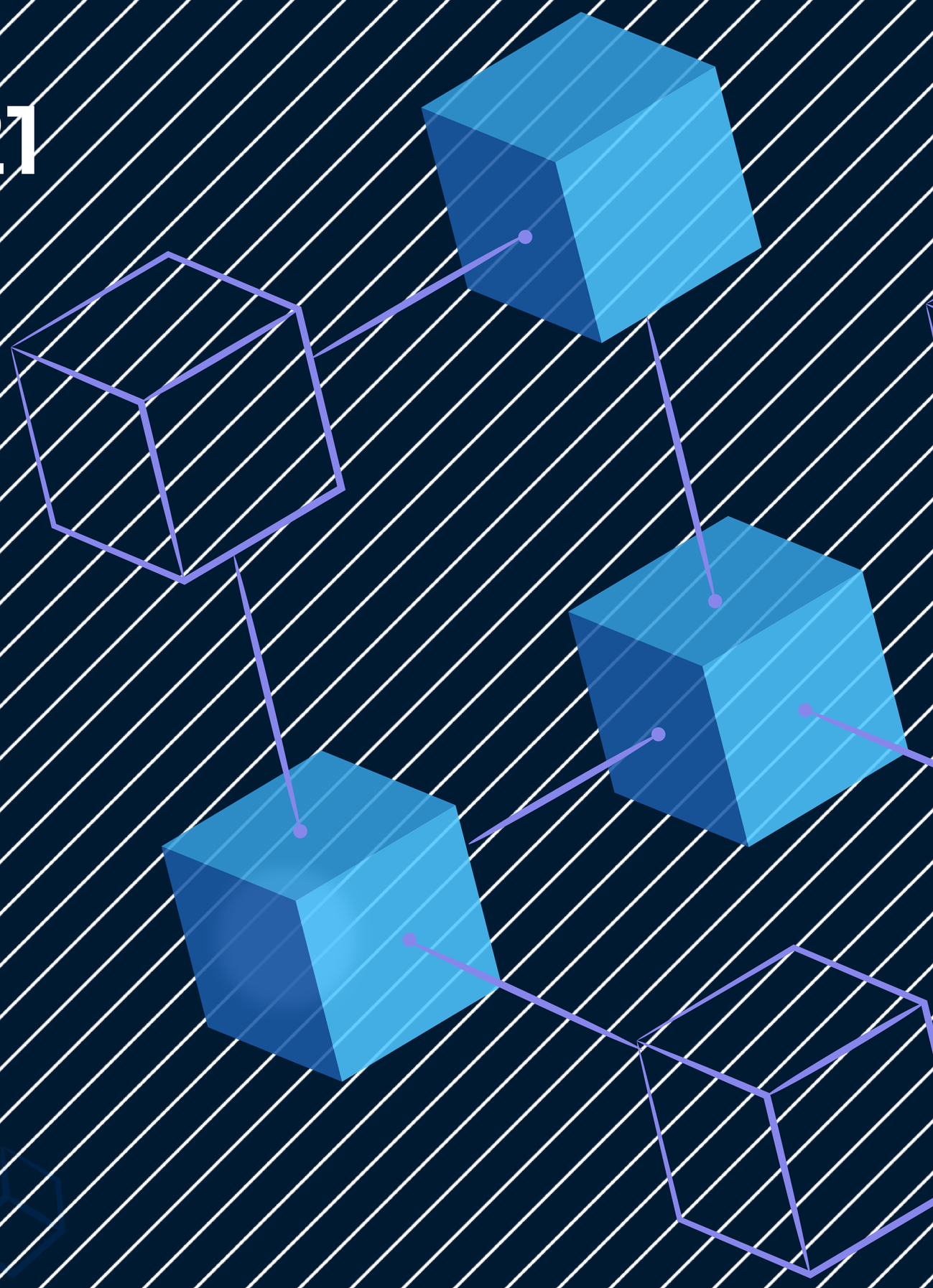
QuillAudits

Audit Report
December, 2021



For

Tryvium

Contents

Scope of the Audit

The scope of this audit was to analyze and document the Tryvium Token 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
- BEP20 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 BEP-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 analysed 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.

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 the security of smart contracts.

Code Review / Manual Analysis
Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the 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 the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis, Theo.

Issue Categories

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

Risk-level	Description
High	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 be fixed before moving to a live environment.
Medium	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	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 issues that 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	0
Acknowledged	0	0	1	2
Closed	0	0	0	0

Introduction

On **Dec 15, 2021- Dec 17,2021** QuillAudits Team performed a security audit for Tryvium smart contracts.
The code for the audit was taken from following the official link:
https://github.com/tryvium-travels/smart-contracts

V	Date	Commit ID	Files
1	Dec 17	ed01c4d0d616832b798f494db90177f5e4c91792	contracts/*

Issues Found – Code Review / Manual Testing

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low severity issues

1. transfer function in TokenVault is susceptible to reentrancy

Line	Code

146-164	function transfer(
	uint256 _amount,
	address payable _to,
	string calldata _reason
	) external onlyOwner {
	_vaultTransfer(_amount, _to, _reason);
	}

Description
nonReentrant modifier is missing from the transfer function. This doesn't pose any immediate risks with the current implementation, however if the team decides to introduce few new features in the contract, this could open up a reentrancy attack vector.

Remediation
Mark transfer function nonReentrant.

Status: Acknowledged

Informational issues

2. TokenVault & BountyVault imports in TryviumToken can be omitted

Line	Code
23-47	/** * @notice The address in which the Team reserved funds are sent. * They correspond to the % of the supply specified in the whitepaper. */ TokenVault immutable public TEAM_VAULT; /** * @notice The address in which the Bounty reserved funds are sent. * They correspond to the % of the supply specified in the whitepaper. */ BountyVault immutable public BOUNTY_VAULT; /** * @notice The address in which the various Token Sales reserved * funds are sent. * They correspond to the % of the supply specified in the whitepaper. */ TokenVault immutable public SALES_VAULT; /** * @notice The address in which the funds reserved for future * developments are sent. * They correspond to the % of the supply specified in the whitepaper. */ TokenVault immutable public RESERVED_FUNDS_VAULT;

Description

Imports of TokenVault & BountyVault in TryviumToken can be omitted, and TEAM_VAULT, BOUNTY_VAULT, SALES_VAULT & RESERVED_FUNDS_VAULT can be declared as address.

Status: Acknowledged

3. Floating pragma

pragma solidity ^0.8.6;

Description
The contract makes use of the floating-point pragma ^0.8.6. Contracts should be deployed using the same compiler version and flags that were used during the testing process. Locking the pragma helps ensure that contracts are not unintentionally deployed using another pragma, such as an obsolete version that may introduce issues in the contract system.

Status: Acknowledged

TestCases for Functional Testing

Some of the test cases which were part of the functional testing are listed below:

TokenVault.sol

Tokens are transfers successfully on invoking transferTokens() function by owner	PASS
Ether is successfully on invoking transfer() function by owner	PASS
Functions transferTokens() & transfer() reverts if caller is not owner:	PASS
Owner should not be able to reenter when invoking function transfer()	FAIL

BountyVault.sol

Tokens are successfully airdropped on invoking airdropTokens() function by owner	PASS
Function airdropTokens() reverts if caller is not owner	PASS

TryviumToken.sol

Only owner can mint tokens	PASS
Only owner can mint tokens if total supply is less than cap	PASS
Only caller can burn their own tokens	PASS
Other user can burn tokens of caller if they are approved	PASS
TryviumToken is following and functioning in accordance to spec	PASS

Automated Tests

Slither

Reentrancy issue mentioned in Issue #1 was found using Slither. Some false positive errors were reported by the tool. All the other issues have been categorized above according to their level of severity.

Closing Summary

Overall, In audit, there are no critical severity issues associated with token transfers.
No instances of Integer Overflow and Underflow vulnerabilities are found in the contract.
One low severity issue found during Audit,which has been Acknowledged by Client.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the Tryvium-Travels LTD Company. 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 Tryvium Travels team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Audit Report
December, 2021

For

Tryvium

QuillAudits

Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com