HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: TosDis
Date: April 7th, 2021

www.hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for TosDis - Second Review
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Transit Token
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Deployed contracts	ETHTosDisTransit: https://kovan.etherscan.io/address/0x31Ba806C0133E83CdbD1100539BF86A30601909F#contracts BSCTosDisTransit: https://testnet.bscscan.com/address/0xBe7F7819e03bc42925fdC4A1afffD174900ECea6#contracts
Timeline	2 APRIL 2021 – 7 APRIL 2021
Changelog	2 APRIL 2021 – INITIAL AUDIT 7 APRIL 2021 – SECOND REVIEW

Table of contents

Introduction

Hacken OÜ (Consultant) was contracted by TosDis (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on April 7th, 2021.

Scope

The scope of the project Is next smart contracts:
ETHTosDisTransit:
https://kovan.etherscan.io/address/0x31Ba806C0133E83CdbD1100539BF86A30601909F#contracts
BSCTosDisTransit:
https://testnet.bscscan.com/address/0xBe7F7819e03bc42925fdC4A1afffD174900ECea6#contracts

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

| Category	Check Item
Code review	▪ Reentrancy
	▪ Ownership Takeover
	▪ Timestamp Dependence
	▪ Gas Limit and Loops
	▪ DoS with (Unexpected) Throw
	▪ DoS with Block Gas Limit
	▪ Transaction-Ordering Dependence
	▪ Style guide violation
	▪ Costly Loop
	▪ ERC20 API violation
	▪ Unchecked external call
	▪ Unchecked math
	▪ Unsafe type inference
	▪ Implicit visibility level
	▪ Deployment Consistency
	▪ Repository Consistency
	▪ Data Consistency

Functional review	- Business Logics Review - Functionality Checks - Access Control & Authorization - Escrow manipulation - Token Supply manipulation - Asset's integrity - User Balances manipulation - Kill-Switch Mechanism - Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contract is well-secured.

| Insecure	Poor secured	Secured	Well-secured

You are
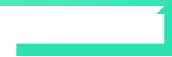
Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. A general overview is presented in AS-IS section, and all found issues can be found in the Audit overview section.

Security engineers found **1** critical, **1** low and **2** informational issues during the first review.

Security engineers found **no issues** during the second review.

Graph 1. The distribution of vulnerabilities after the first review.

- Informational
- Critical
- Low
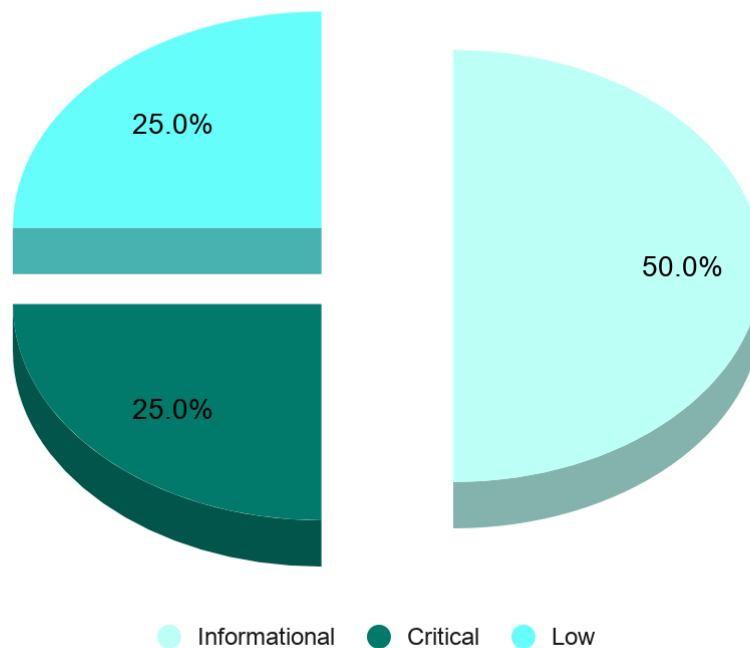
Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations, and info statements can't affect smart contract execution and can be ignored.

This document is proprietary and confidential. No part of this document may be disclosed
in any manner to a third party without the prior written consent of Hacken.

www.hacken.io

Audit overview

■■■■ Critical

1. **Vulnerability:** Reentrancy
   Contract: ETHTosDisTransit, BSCTosDisTransit
   Method: collectFee()

   Fixed before second review

■ ■ ■ High

No High severity issues were found.

■■ Medium

No Medium severity issues were found.

■ Low

1. **Vulnerability:** Missing events arithmetic
   Contract: ETHTosDisTransit, BSCTosDisTransit

   Fixed before second review

■ Lowest / Code style / Best Practice

1. **Vulnerability:** Boolean equality
   Contract: ETHTosDisTransit, BSCTosDisTransit

   Fixed before second review

2. Lines 275, 307, 320 of the ETHTosDisTransit.sol and 350, 378, 391 and
   410 of BSCTosDisTransit.sol are above the recommended <u>maximum line
   length</u>.

   Fixed before second review

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. For the contract, high-level description of functionality was presented in As-Is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security engineers found **1** critical, **1** low and **2** informational issues during the first review.

Security engineers found **no issues** during the second review.

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. Thus, the audit can't guarantee the explicit security of the audited smart contracts.