HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Dios Finance
Date: January 12th, 2022

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for Dios Finance.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Staking
Platform	Binance Smart Chain / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Deployed contract	https://bscscan.com/address/0x36c8a6E7436EDd850752E09539a519a369D95096#code
Technical Documentation	YES
JS tests	NO
Website	https://dios.finance/
Timeline	29 DECEMBER 2021 - 12 JANUARY 2022
Changelog	12 JANUARY 2022 - INITIAL AUDIT

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by Dios Finance (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contract and its code review conducted between December 29th, 2021 - January 12th, 2022.

Scope

The scope of the project is smart contracts in the blockchain:

Explorer URL:
https://bscscan.com/address/0x36c8a6E7436EDd850752E09539a519a369D95096#code
Technical Documentation: Yes (https://docs.dios.finance/info/staking)
JS tests: No
Contracts:
Staking.sol

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

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency manipulation ▪ Kill-Switch Mechanism ▪ Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contracts are well-secured.

Insecure	Poor secured	Secured	Well-secured

You are here

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. All found issues can be found in the Audit overview section.

As a result of the audit, security engineers found **3** low severity issues.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

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

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Audit overview

Critical

No critical issues were found.

High

No high severity issues were found.

Medium

No medium severity issues were found.

Low

1. Code inconsistency to the documentation.

   The provided documentation says:

   We will have a tax for swapping from sDIOS to DIOS, Tax will be used for Token Burning and Treasury. Tax rate will start at 25% and drop off 1% per day (00:00 UTC + 0), calculating from the date Staking started. The Tax rate is fixed at 4% after 21 days from Staking start.

   While in the code there is a modifier “checkTax” which changes the tax once per 28800 blocks, which is not “00:00 UTC + 0”

   Contract: Staking.sol

   Modifier: checkTax

   Recommendation: While the code is already deployed and the tax decreasing period is over, no actions could be done. But in case you’re planning to reuse the staking contract, please make sure that docs and the code are consistent.

2. Tautology or contradiction.

   _taxRate is a **uint256**, so **_taxRate >= 0** will be always **true**.

   Contract: Staking.sol

   Modifier: setTaxRate

   Recommendation: While the code is already deployed no actions could be done. But for the latter: fix the incorrect comparison by changing the value type or the comparison.

3. Events absence.

   Any changes in the contract states (taxes, bonuses, stakes, unstakes, etc.) are recommended to follow up with emitting events. That’s very

useful to the community and provides an incredible way to track the contract off-chain.

Contract: Staking.sol

Functions: stake, claim, forfeit, toggleDepositLock, unstake, giveLockBonus, returnLockBonus, setContract, setTaxRate, setDAO, setDAO

Recommendation: While the code is already deployed no actions could be done. But for the latter: please make sure you're emitting events on the contract state changes.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools.

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

As a result of the audit, security engineers found **3** low severity issues.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

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 the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bug-free 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 the security of smart contracts.

Technical Disclaimer

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