HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Nimbus
Date: May 30th, 2021

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 Nimbus.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Token, Referral Program
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/nimbusplatformorg/nim-smartcontract/tree/7bda71190cca5d139e15b46a33ca041eb060f38d (Initial Audit) https://github.com/nimbusplatformorg/nim-smartcontract/commit/6e57eafcdc7b9a08ccb0369bf135a69ce4680be5 (Remediation)
Deployed contract	
Changelog	03 May 2021 - Initial Audit 30 May 2021 - remediation

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 Nimbus (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 May 3rd, 2021. Remediation conducted on May 30th, 2021.

Scope

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

```
Repository: https://github.com/nimbusplatformorg/nim-smartcontract/
Commit: 6e57eafcdc7b9a08ccb0369bf135a69ce4680be5
Files:
      NimbusCore/NBU.sol
      NimbusCore/NimbusReferralProgram.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
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

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

Executive Summary

According to the assessment, the Customer's smart contracts are secure.

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.

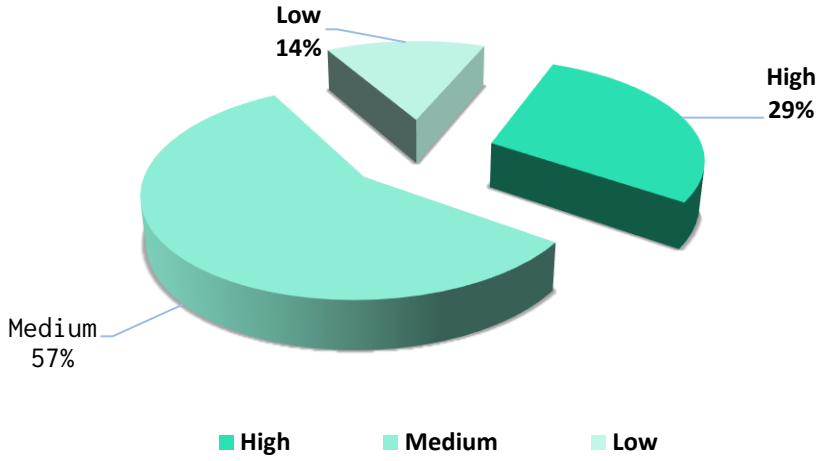
As a result of the audit, security engineers found 2 high, 4 medium and 1 low severity issues.

After the second review, the code contains no issues.

Notices:

1. Description of contracts logic is not provided by the Customer, and we may not prove correctness of calculation.

Graph 1. The distribution of vulnerabilities after the audit.



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

1. The function returns frozen balances in addition to active ones. The transfer function can fail if a user decides to transfer all his tokens but has unclaimed vested tokens.

   Contract: NBU.sol

   Function: balanceOf

   Recommendation: separate vesting balance from active balance.

   Status: the Customer approved that such behavior is intended.

2. The function sums balances of different tokens and proceeds calculations with those values.

   Contract: NimbusReferralProgram.sol

   Function: isUserBalanceEnough

   Recommendation: balances of different tokens should be cared separately.

   Status: the Customer approved that such behavior is intended.

Medium

1. Vesting types are not validated. Any number can be passed.

   Contract: NBU.sol

   Function: give

   Recommendation: validate vesterId parameter.

   Status: Addressed in 12A41A194F39670637D79EF2C5BCC6A70D617781 commit.

2. The function returns 0 address for all users whose sponsorId is lower than sponsorId.

   Contract: NimbusReferralProgram.sol

   Function: userSponsorAddressByAddress

   Recommendation: fix if statement.

Status: Addressed in 12A41A194F39670637D79EF2C5BCC6A70D617781 commit.

3. The function does not actually transfer anything for sponsors with id < 1000000001.

   Contract: NimbusReferralProgram.sol

   Function: transferToSponsor

   Status: not an issue.

4. The function returns zero address for all sponsors with is < 1000000001. Such id could not be reached.

   Contract: NimbusReferralProgram.sol

   Function: userSponsorAddressByAddress

   Status: the Customer approved that such behavior is intended.

- Low

1. The SafeMath library is redundant for compiler versions >= 8.0.0. All operations upon uint data type are checked.

   Contracts: all

   Recommendation: remove redundant libraries.

   Status: Addressed in 12A41A194F39670637D79EF2C5BCC6A70D617781 commit.

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.

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

As a result of the audit, security engineers found 2 high, 4 medium and 1 low severity issues.

After the second review, the code contains no issues.

Notices:

1. Description of contracts logic is not provided by the Customer, and we may not prove correctness of calculation.

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 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.