# SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Nimbus
Date: May 30th, 2021

This document may contain confidential information about IT systems and the intellectual property of the Customer as well as information about potential vulnerabilities and methods of their exploitation.

The report containing confidential information can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities fixed – upon a decision of the Customer.

## Document

| | |
|---|---|
| Name | Smart Contract Code Review and Security Analysis Report for Nimbus. |
| Approved by | Andrew Matiukhin \| CTO Hacken OU |
| Type | Governance, Token |
| Platform | Ethereum / Solidity |
| Methods | Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review |
| Repository | https://github.com/nimbusplatformorg/nim-smartcontract/tree/7bda71190cca5d139e15b46a33ca041eb060f38d (INITIAL AUDIT)<br>https://github.com/nimbusplatformorg/nim-smartcontract/commit/6e57eafcdc7b9a08ccb0369bf135a69ce4680be5 (REMEDIATION) |
| Deployed contract | |
| Changelog | 10 MAY 2021 - INITIAL AUDIT<br>30 MAY 2021 - REMEDIATION |

## Table of contents

## 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 10th, 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:
Governance/GNBU.sol
Governance/NimbusGovernorV1.sol

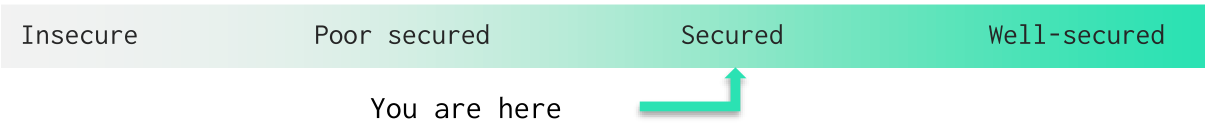
We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

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

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

## Executive Summary

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



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.

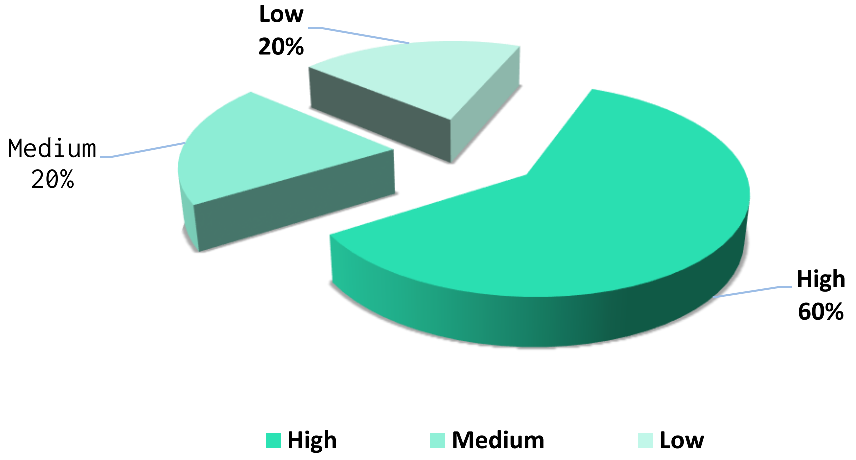
As a result of the audit, security engineers found 3 high, 1 medium and 1 low severity issues.

After the second review, the contracts contain 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.*



## 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 |

## Audit overview

🟥🟥🟥🟥 Critical

🟨🟨🟨 High

1. Limiting approve and transfer amount to 96 bits violates the ERC-20 standard. This can lead to limited acceptance of the asset and fund losses in some rare cases.

   Contract: GNBU.sol

   Function: approve, permit

   Recommendation: allow usage of uint256 values or cast all values that exceeds 96 bits to max uint96 values.

   Status: the Customer approved that such behavior is intended.

2. 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: GNBU.sol

   Function: balanceOf

   Recommendation: separate vesting balance from active balance.

   Status: the Customer approved that such behavior is intended.

3. Proposal can be cancelled only if a proposer has not enough votes and not enough stake balance. This may lead to execution of unsafe proposals.

   Contract: NimbusGovernorV1.sol

   Function: cancel

   Recommendation: add administrative role that can cancel unsafe proposals.

   Status: the Customer approved that such behavior is intended.

🟩🟩 Medium

1. Custom calculation of uint max values are used.

   Contract: GNBU.sol

   Recommendation: use type(T).max instead.

Status: Addressed in 6E57EAFCDC7B9A08CCB0369BF135A69CE4680BE5 commit.

- 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 6E57EAFCDC7B9A08CCB0369BF135A69CE4680BE5 commit.

## 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 3 high, 1 medium and 1 low severity issues.

After the second review, the contracts contain no issues.

Notices:

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

## 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.