



HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Nitro Network

Date: January 29th, 2022

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 are fixed – upon a decision of the Customer.

Document

Name	Smart Contract Code Review and Security Analysis Report for Nitro Network.
Approved by	Andrew Matiukhin CTO Hacken OU
Type	ERC20 token;
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/NucleusVision/Smart-Contract-2.0
Commit	355496d716E2822d75A1B15B1A5FE62D86119c14
Technical Documentation	NO
JS tests	NO
Website	nitro.network
Timeline	06 JANUARY 2022 - 29 JANUARY 2022
Changelog	11 JANUARY 2022 - INITIAL AUDIT 20 JANUARY 2022 - SECOND AUDIT 29 JANUARY 2022 - THIRD AUDIT



Table of contents

Introduction	4
Scope	4
Executive Summary	5
Severity Definitions	8
Audit overview	9
Conclusion	11
Disclaimers	12

Introduction

Hacken OÜ (Consultant) was contracted by Nitro Network (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 January 6th, 2022 - January 11th, 2022.

The second review was conducted on January 20th, 2022.

The third review was conducted on January 29th, 2022.

Scope

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

Repository:

<https://github.com/NucleusVision/Smart-Contract-2.0>

Commit:

[355496d716e2822d75a1b15b1a5fe62d86119c14](#)

Technical Documentation: No

JS tests: No

Contracts:

[./contracts/NitroNetwork.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	<ul style="list-style-type: none"> ▪ 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	<ul style="list-style-type: none"> ▪ 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.



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 **4** critical and **2** low severity issues.

As a result of the second review, security engineers found that **2** low severity issues were resolved, while **4** critical issues remained.

As a result of the third review, security engineers found **no** severity issues, all previously found issues were resolved.

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

1. The mint function allows the MINTER_ROLE mint tokens without any restrictions. It can lead to token supply manipulations.

Contracts: NitroNetwork.sol

Function: mint

Recommendation: provide the MINTER_ROLE contract with clear minting rules, or remove the possibility of unlimited minting.

Status: fixed

2. The burnBlackFunds function allows OPERATOR_ROLE to burn all tokens of a user who was added to the blacklist by the OPERATOR_ROLE.

Contracts: NitroNetwork.sol

Function: burnBlackFunds

Recommendation: remove the possibility to burn users' tokens.

Status: fixed

3. All the token transfers can be stoped by the PAUSER_ROLE.

Contracts: NitroNetwork.sol

Function: _beforeTokenTransfer

Recommendation: provide the PAUSER_ROLE contract with clear pause rules, or remove the possibility to stop transfers.

Status: fixed

4. All user transfers can be blocked by adding this user to the blacklist by OPERATOR_ROLE.

Contracts: NitroNetwork.sol

Function: _beforeTokenTransfer



Recommendation: remove the possibility to block users' token transfers.

Status: fixed

■ ■ ■ High

No high severity issues were found.

■ ■ Medium

No medium severity issues were found.

■ Low

1. The `_mintTo` parameter of the `initialize` function is unused.

Contracts: NitroNetwork.sol

Function: initialize

Recommendation: remove the unused parameter.

Status: fixed

2. The `GOVERNANCE_ROLE` role has been configured but not in use.

Contracts: NitroNetwork.sol

Function: initialize

Recommendation: remove the unused role.

Status: fixed

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 **4** critical and **2** low severity issues.

As a result of the second review, security engineers found that **2** low severity issues were resolved, while **4** critical issues remained.

As a result of the third review, security engineers found **no** severity issues, all previously found issues were resolved.

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.