

**HACKEN**

# SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

**Customer:** SDAO

**Date:** March 10<sup>th</sup>, 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

<b>Name</b>	Smart Contract Code Review and Security Analysis Report for SDAO.
<b>Approved by</b>	Andrew Matiukhin   CTO Hacken OU Evgeniy Bezuglyi   SC Department Head at Hacken OU
<b>Type</b>	ERC20 Converter
<b>Platform</b>	EVM
<b>Language</b>	Solidity
<b>Methods</b>	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
<b>Repository</b>	<a href="https://gitlab.com/nunet/upgradable-tokens">https://gitlab.com/nunet/upgradable-tokens</a>
<b>Commit</b>	c4c0b77524df3890221269226bc13b2d6da08d4f
<b>Technical Documentation</b>	NO
<b>JS tests</b>	YES
<b>Website</b>	<a href="https://singularitydao.ai/">https://singularitydao.ai/</a>
<b>Timeline</b>	04 MARCH 2022 - 10 MARCH 2022
<b>Changelog</b>	10 MARCH 2022 - INITIAL AUDIT



## Table of contents

Introduction	4
Scope	4
Executive Summary	5
Severity Definitions	7
Audit overview	8
Recommendations	9
Disclaimers	10

## Introduction

Hacken OÜ (Consultant) was contracted by SDAO (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 contracts.

## Scope

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

**Repository:**

<https://gitlab.com/nunet/upgradable-tokens>

**Commit:**

[c4c0b77524df3890221269226bc13b2d6da08d4f](#)

**Technical Documentation:** No

**JS tests:** Yes; in tests directory

**Contracts:**

[NuNetUpgradeableToken.sol](#)  
[NuNetUpgradeableTokenV2.sol](#)  
[NuNetUpgradeableTokenV3.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"><li>▪ Reentrancy</li><li>▪ Ownership Takeover</li><li>▪ Timestamp Dependence</li><li>▪ Gas Limit and Loops</li><li>▪ Transaction-Ordering Dependence</li><li>▪ Style guide violation</li><li>▪ EIP standards violation</li><li>▪ Unchecked external call</li><li>▪ Unchecked math</li><li>▪ Unsafe type inference</li><li>▪ Implicit visibility level</li><li>▪ Deployment Consistency</li><li>▪ Repository Consistency</li></ul>
Functional review	<ul style="list-style-type: none"><li>▪ Business Logics Review</li><li>▪ Functionality Checks</li><li>▪ Access Control &amp; Authorization</li><li>▪ Escrow manipulation</li><li>▪ Token Supply manipulation</li><li>▪ Assets integrity</li><li>▪ User Balances manipulation</li><li>▪ Data Consistency</li><li>▪ Kill-Switch Mechanism</li></ul>

## Executive Summary

Score measurements details can be found in the corresponding section of the [methodology](#).

### Documentation quality

The customer has neither provided any functional nor technical requirements.

Total Documentation Quality score is **0** out of **10**.

### Code quality

Total CodeQuality score is **6** out of **10**. Too long lines. No DocBlocks. Spacing issues.

### Architecture quality

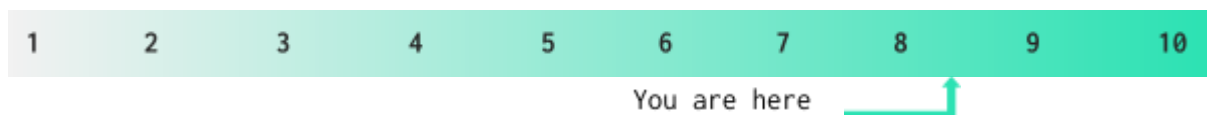
Architecture quality score is **8** out of **10**. The outdated compiler version is used. Some functions could be declared pure, some external.

### Security score

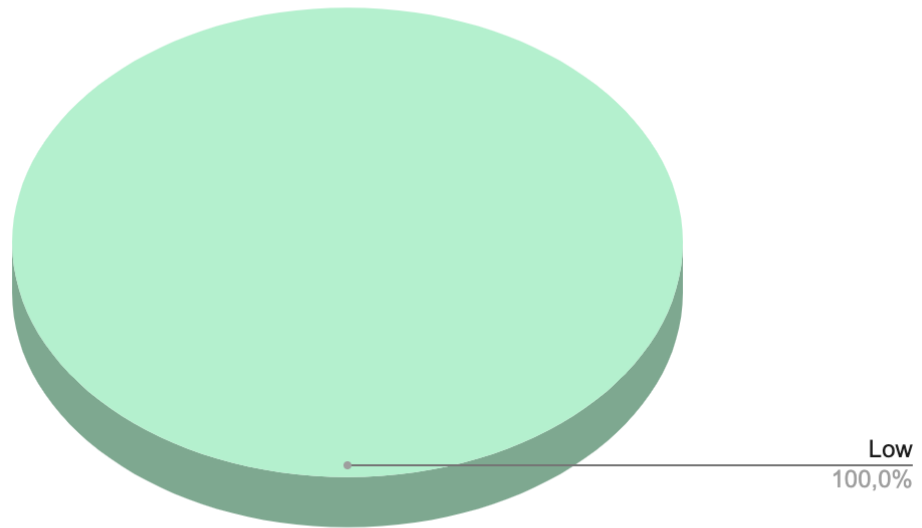
As a result of the audit, security engineers found **1** low severity issue. Security score is **10** out of **10**. All found issues are displayed in the “Issues overview” section of the report.

### Summary

According to the assessment, the Customer's smart has the following score: **8.4**



*Graph 1. The distribution of vulnerabilities after the audit.*



## Severity Definitions

Risk Level	Description
<b>Critical</b>	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
<b>High</b>	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
<b>Medium</b>	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
<b>Low</b>	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution

## Audit overview

### ■■■■ Critical

No critical issues were found.

### ■■■ High

No high severity issues were found.

### ■■ Medium

No medium severity issues were found.

### ■ Low

1. A public function that could be declared external

**public** functions that are never called by the contract should be declared **external**.

**Contract:** NuNetUpgradeableToken.sol, NuNetUpgradeableTokenV2.sol, NuNetUpgradeableTokenV3.sol

**Functions:** decimals, burn, additionalMint

**Recommendation:** Use the **external** attribute for functions never called from the contract.

**Status:** New

2. A view function that could be declared pure

**view** functions that are neither access contract state nor do external calls should be declared **pure** to save gas.

**Contract:** NuNetUpgradeableToken.sol, NuNetUpgradeableTokenV2.sol, NuNetUpgradeableTokenV3.sol

**Functions:** decimals

**Recommendation:** Use the **pure** attribute for functions never access state.

**Status:** New





## Recommendations

1. Please follow solidity recommendations for the code formatting.
2. To minimize gas usages try to use pure functions, instead of view.
3. Try using a more modern compiler version, e.g. 0.8.11

**Contracts:** NuNetUpgradeableToken.sol, NuNetUpgradeableTokenV2.sol,  
NuNetUpgradeableTokenV3.sol



## 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.