

HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Gamestarter

Date: August 17th, 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 are fixed – upon a decision of the Customer.

Document

Name	Smart Contract Code Review and Security Analysis Report for Gamestarter.
Approved by	Andrew Matiukhin CTO Hacken OU
Type	ERC20 token; Vesting Schedule
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Zip archive	contracts-ae459361588f564135284510531f11ee3c3b2c78.zip
Technical Documentation	NO
JS tests	YES
Timeline	28 JULY 2021 - 17 AUGUST 2021
Changelog	03 AUGUST 2021 - INITIAL AUDIT 17 AUGUST 2021 - SECOND REVIEW



Table of contents

Introduction	4
Scope	4
Executive Summary	5
Severity Definitions	7
Audit overview	8
Conclusion	9
Disclaimers	11



Introduction

Hacken OÜ (Consultant) was contracted by Gamestarter (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 July 28th, 2021 - August 3rd, 2021. The second review conducted on August 17th, 2021.

Scope

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

Zip archive:

[contracts-ae459361588f564135284510531f11ee3c3b2c78.zip](#)

Commit:

[4d794ff846e30477e159937186ffac2](#)

Technical Documentation: No

JS tests: Yes

Contracts:

[GameCoin.sol](#)

[GameCoinVestingSchedule.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 secured.



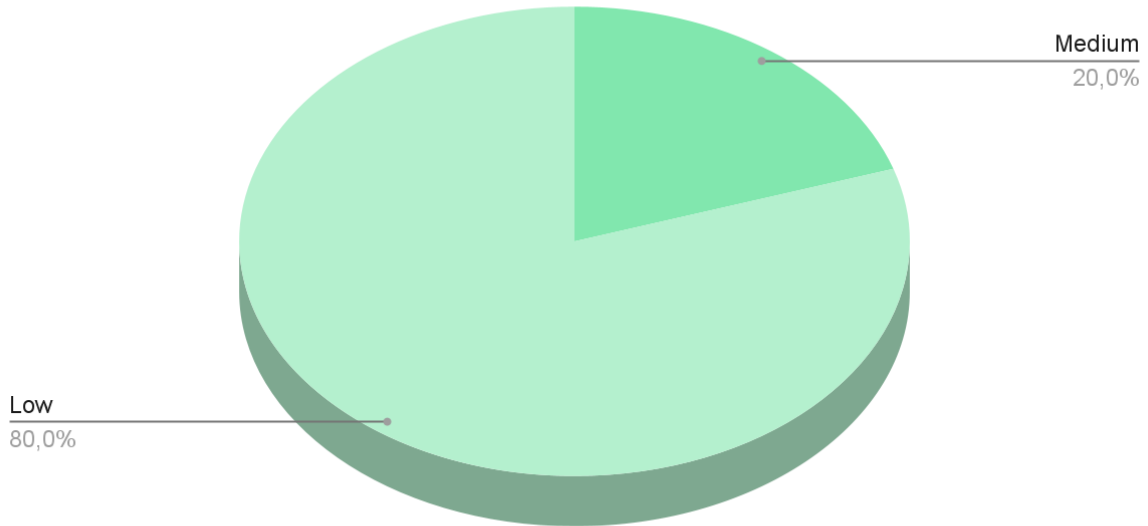
You are here 

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.

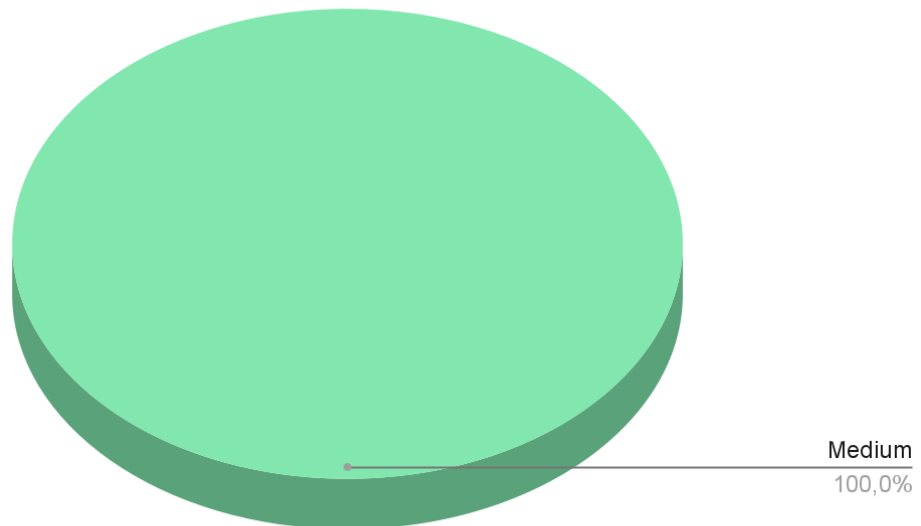
As a result of the audit, security engineers found **1** medium and **3** low severity issue.

After the second review security engineers found only **1** medium issue with tests, all low severity issues were fixed before the second review.

Graph 1. The distribution of vulnerabilities after the audit.



Graph 2. The distribution of vulnerabilities after the second review.





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

No critical issues were found.

■ ■ ■ High

No high severity issues were found.

■ ■ Medium

Little test coverage

Tests provided cover only a few cases and also have a lot of TODOs in those.

Recommendation: Please test more cases. The best practice is to have at least 95% code coverage.

Second Review: Added more tests. 13 of 59 tests are failing. Please consider re-checking the business logic in the code or tests.

Customer's comment: Those 13 tests are supposed to fail (marked (Should Fail) next to test). They are admin rights checks etc.

■ Low

1. Unused import.

Both GameCoinVestingSchedule.sol and GameCoin.sol have import statements for "hardhat/console.sol" which is never used in the contracts.

Recommendation: Please remove or comment out those imports.

Fixed before the second review.

2. Too many digits.

Literals with many digits are difficult to read and review.

Recommendation: While provided token declared as 5 digits, we'd suggest declaring 100k of tokens like one of the following:

- 1e13
- 100e11
- 100_000_000e5
- 100e6 * 1e5

Fixed before the second review.

3. A public function that could be declared external

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

Fixed before the second review.



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 **1** medium and **4** low severity issue.

After the second review security engineers found only **1** medium issue with tests, all low severity issues were fixed before the second review.



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.