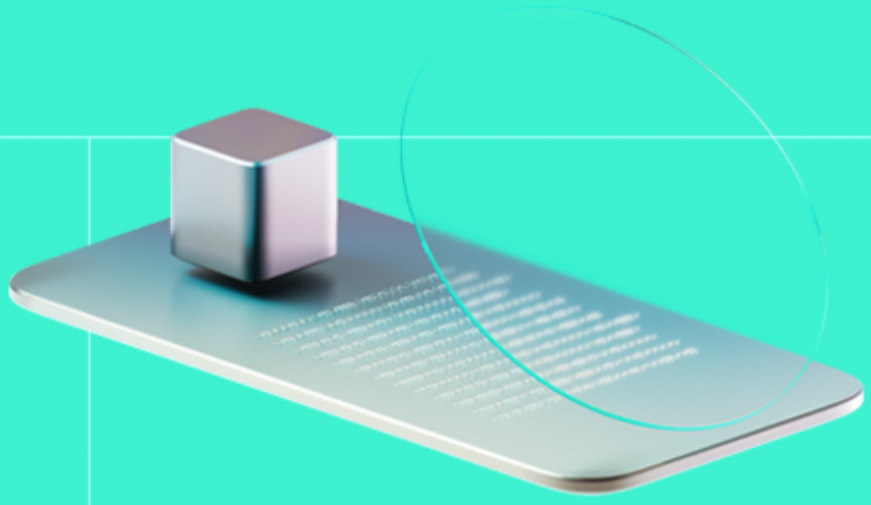
HACKEN

Smart Contract Code Review And Security Analysis Report



Customer: Paycheck

Date: 02/01/2025

We express our gratitude to the Paycheck team for the collaborative engagement that enabled the execution of this Smart Contract Security Assessment.

Paycheck is a platform that integrates blockchain and AI to offer financial solutions. One of its key features is the issuance of $CHECK token, enabling NFT checks.

Document

Name	Smart Contract Code Review and Security Analysis Report for Paycheck
Audited By	Adam Idarrha
Approved By	Oleksii Haponiuk
Website	https://paycheck.io/
Changelog	30/12/2024 - Preliminary Report, 02/01/2025 - Final Report
Platform	Polygon
Language	Solidity
Tags	Fungible Token, Permit Token
Methodology	https://hackenio.cc/sc_methodology

Review Scope

Repository	https://github.com/PaycheckLabs/contracts
Commit	4b8ed98

Audit Summary

The system users should acknowledge all the risks summed up in the risks section of the report

0	0	0	0
Total Findings	Resolved	Accepted	Mitigated

Findings by Severity

Severity	Count
Critical	0
High	0
Medium	0
Low	0

Documentation quality

- Functional requirements are provided.
- Technical description is provided.

Code quality

- The code mostly conforms with the Solidity Style Guide.
- The development environment is configured.

Test coverage

Code coverage of the project is **100%** (branch coverage).

- Deployment and basic user interactions are covered with tests.

Table of Contents

System Overview

Check Token — simple ERC-20 token that mints all initial supply to a deployer. It features burnable capabilities and permit functionality, with no option for additional minting.

It has the following attributes:

- Name: Check Token
- Symbol: CHECK
- Decimals: 18
- Total supply: 100 billion tokens.

Potential Risks

- The contract mints the entire token supply to the deployer upon deployment, creating a significant centralization risk. This concentration of tokens in a single address gives the deployer disproportionate control over the token's distribution and potentially the protocol's governance. However, according to the Paycheck team, this approach is part of the token migration process and is managed using a multi-signature wallet, mitigating concerns about long-term centralization.

Findings

Vulnerability Details

Observation Details

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed based on best industry practices at the time of the writing of this report, with 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 report contains no statements or warranties on the identification of all vulnerabilities and security of the code. The report covers the code submitted and reviewed, so it may not be relevant after any modifications. Do not consider this report as a final and sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements.

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.

English is the original language of the report. The Consultant is not responsible for the correctness of the translated versions.

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 Consultant cannot guarantee the explicit security of the audited smart contracts.

Appendix 1. Definitions

Severities

When auditing smart contracts, Hacken is using a risk-based approach that considers **Likelihood**, **Impact**, **Exploitability** and **Complexity** metrics to evaluate findings and score severities.

Reference on how risk scoring is done is available through the repository in our Github organization:

hknio/severity-formula

Severity	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to the loss of user funds or contract state manipulation.
High	High vulnerabilities are usually harder to exploit, requiring specific conditions, or have a more limited scope, but can still lead to the loss of user funds or contract state manipulation.
Medium	Medium vulnerabilities are usually limited to state manipulations and, in most cases, cannot lead to asset loss. Contradictions and requirements violations. Major deviations from best practices are also in this category.
Low	Major deviations from best practices or major Gas inefficiency. These issues will not have a significant impact on code execution.

Potential Risks

The "Potential Risks" section identifies issues that are not direct security vulnerabilities but could still affect the project’s performance, reliability, or user trust. These risks arise from design choices, architectural decisions, or operational practices that, while not immediately exploitable, may lead to problems under certain conditions. Additionally, potential risks can impact the quality of the audit itself, as they may involve external factors or components beyond the scope of the audit, leading to incomplete assessments or oversight of key areas. This section aims to provide a broader perspective on factors that could affect the project's long-term security, functionality, and the comprehensiveness of the audit findings.

Appendix 2. Scope

The scope of the project includes the following smart contracts from the provided repository:

Scope Details	
Repository	https://github.com/PaycheckLabs/contracts/
Commit	4b8ed98736ce7b83db9a570f747d326fa2688a6e
Whitepaper	
Requirements	
Technical Requirements	

Asset	Type
CheckToken.sol [https://github.com/PaycheckLabs/contracts/]	Smart Contract