

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



Customer: Trickle Technologies Inc

Date: November 15th, 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		
	Trickle Technologies Inc.		
	9		
Approved by	Andrew Matiukhin CTO Hacken OU		
Туре	ERC20 token; Transfer controller		
Platform	Ethereum / Solidity		
Methods	Architecture Review, Functional Testing, Computer-Aided		
	Verification, Manual Review		
Repository	https://github.com/Trickle-Cloud/token		
Commit	f4942c20b873927436a09cd5137d751f0fab1f86		
Technical	NO		
Documentation	1.5		
Documentation			
JS tests	NO		
Website	trickle.cloud		
Timeline	10 NOVEMBER 2021 - 15 NOVEMBER 2021		
Changelog	12 NOVEMBER 2021 - Initial Audit		
	15 NOVEMBER 2021 - Second Review		
	13 NOTERIDEN 2021 SCOOLS NOTEW		

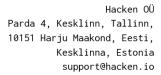




Table of contents

Introduction	4
Scope	4
Executive Summary	5
Severity Definitions	6
Audit overview	7
Conclusion	8
Disclaimers	9



Introduction

Hacken OÜ (Consultant) was contracted by Trickle Technologies Inc (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 November $10^{\rm th}$, 2021 - November $12^{\rm th}$, 2021.

Second review conducted on November 15th, 2021.

Scope

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

Repository:

https://github.com/Trickle-Cloud/token

Commit:

f4942c20b873927436a09cd5137d751f0fab1f86

Technical Documentation: No

JS tests: No Contracts:

Trickle.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	
	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 	



_		
Funct	ional	review

- 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.

Insecure	Poor secured	Secured	Well-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 high and 1 low severity issue.

As a result of the second review, security engineers found no issue.



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

All token transfers can be stopped by owners. Such functionality can be used to manipulate the market

Contracts: Trickle.sol

Functions: pause

Recommendation: The owner must be a contract with transparent rules for using the pause function. Else remove this function.

Status: fixed.

■ Medium

No medium severity issues were found.

Low

"hardhat/console.sol" is imported but never used

Contracts:Trickle.sol

Recommendation: remove unused import.

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 ${\bf 1}$ high and ${\bf 1}$ low severity issue.

As a result of the second review, security engineers found **no** issue.



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.