

HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Lugh

Date: March 16th, 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 Lugh.
Approved by	Andrew Matiukhin CTO Hacken OU
Type	ERC20 token; Transfer controller
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://gitlab.com/sceme/eurl-solidity
Commit	?????
Deployed contract	-
Technical Documentation	YES
JS tests	YES
Website	https://www.lugh.io/
Timeline	15 FEBRUARY 2022 - 16 MARCH 2022
Changelog	17 FEBRUARY 2022 - INITIAL AUDIT 16 MARCH 2022 - SECOND REVIEW



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 Lugh (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 February 15th, 2022 - February 17th, 2022.

Scope

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

Repository:

<https://gitlab.com/sceme/eurl-solidity>

Commit:

??????

Technical Documentation: Yes

- specs_eth_2021-02-10.pdf (md5: 0669a2542e62df451cabe5c70fea5221)

JS tests: Yes

- included in the "test" directory

Contracts:

[Blacklistable.sol](#)

[Forwarder.sol](#)

[Migrations.sol](#)

[Token.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 analyzed 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** low severity issue.

After the second review, security engineers found that contracts were slightly changed by renaming variables, new events were added, and the Forwarder contract became “OwnableUpgradeable” instead of “Initializable and ContextUpgradeable”. Therefore **no security issues** were found.

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

No medium severity issues were found.

■ Low

1. No events on changing values.

It is recommended to emit events when changing crucial contract states to allow the community to track such changes off-chain.

Contract: Token.sol

Functions: setFeeFaucet, updateTxFeeRate, updateGaslessBasefee, setTrustedForwarder

Recommendation: Please emit events on changing important contract states.

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 **1** low severity issue.

After the second review, security engineers found that contracts were slightly changed by renaming variables, new events were added, and the Forwarder contract became “OwnableUpgradeable” instead of “Initializable and ContextUpgradeable”. Therefore **no security issues** were found.



Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date 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 audit makes no statements or warranties on the security of the code. It also cannot be considered a 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 it 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 cannot guarantee the explicit security of the audited smart contracts.