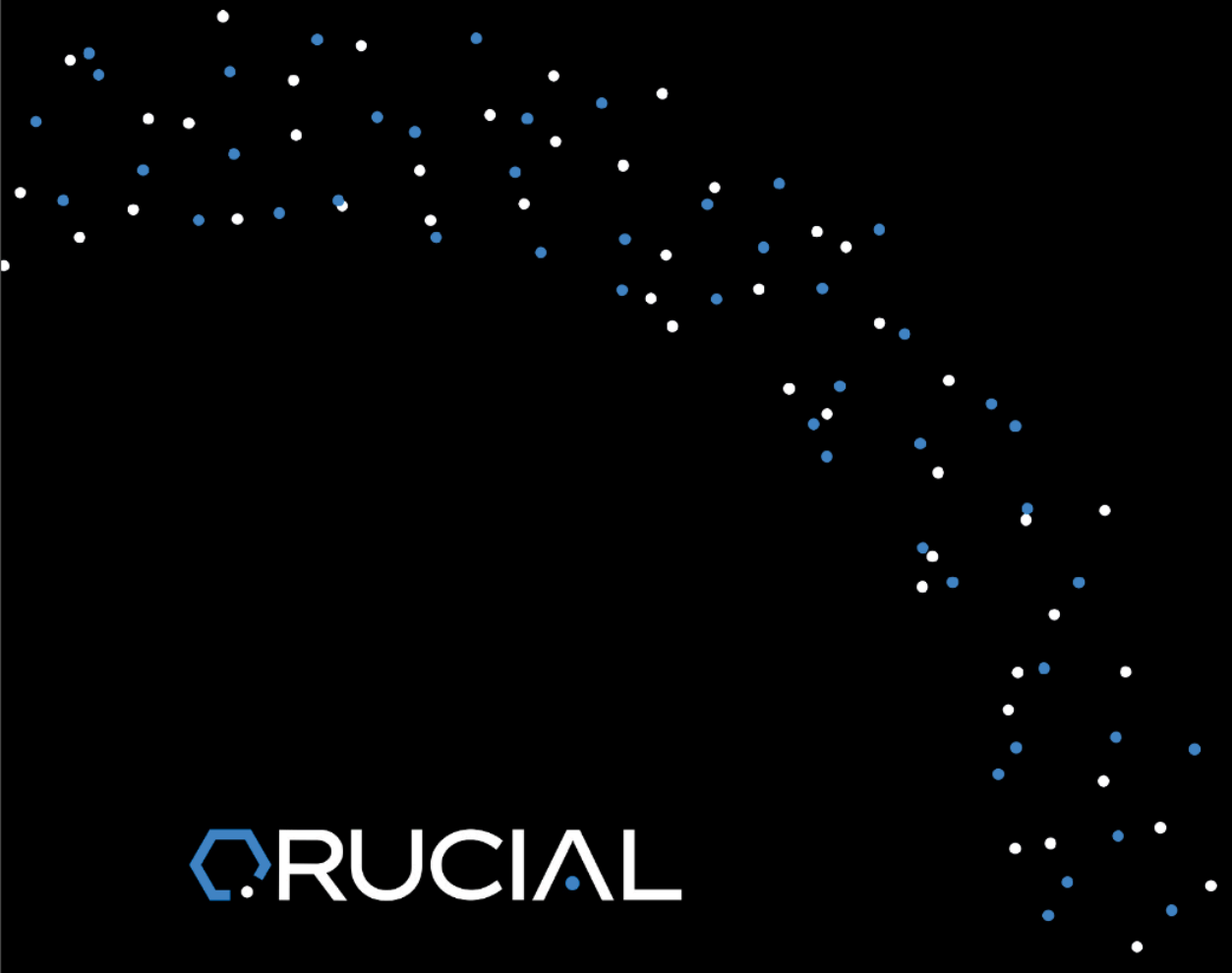
SMART CONTRACT SECURITY REPORT.

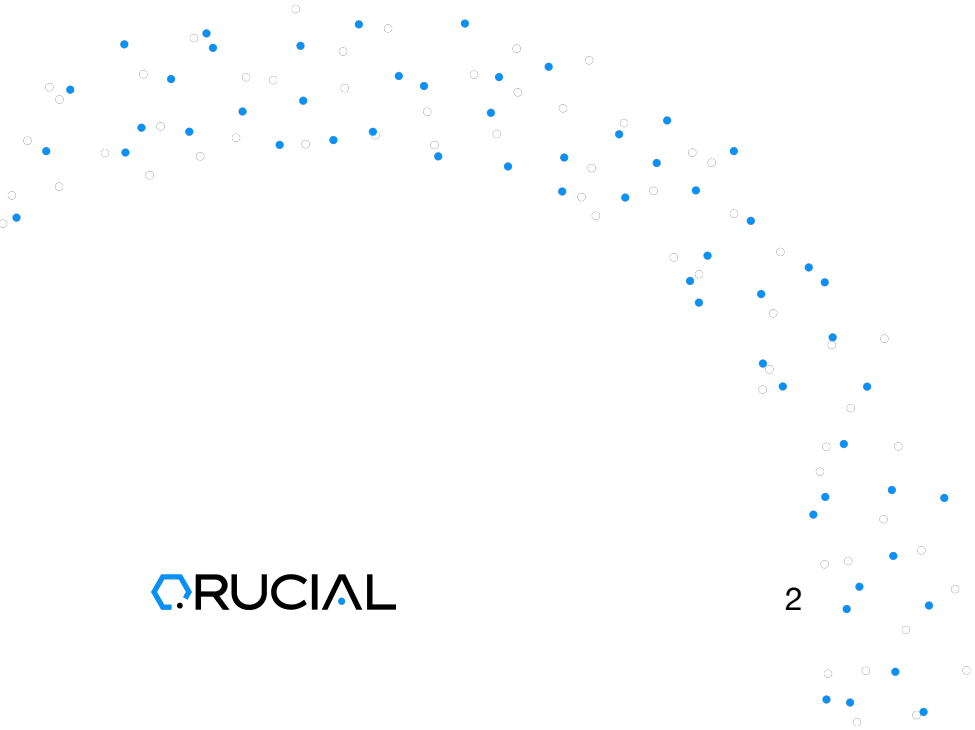
Personally audited and created for:
ink! ERC20 implementation by Parity



CRUCIAL

10th of August 2022

TABLE OF CONTENTS



Introduction

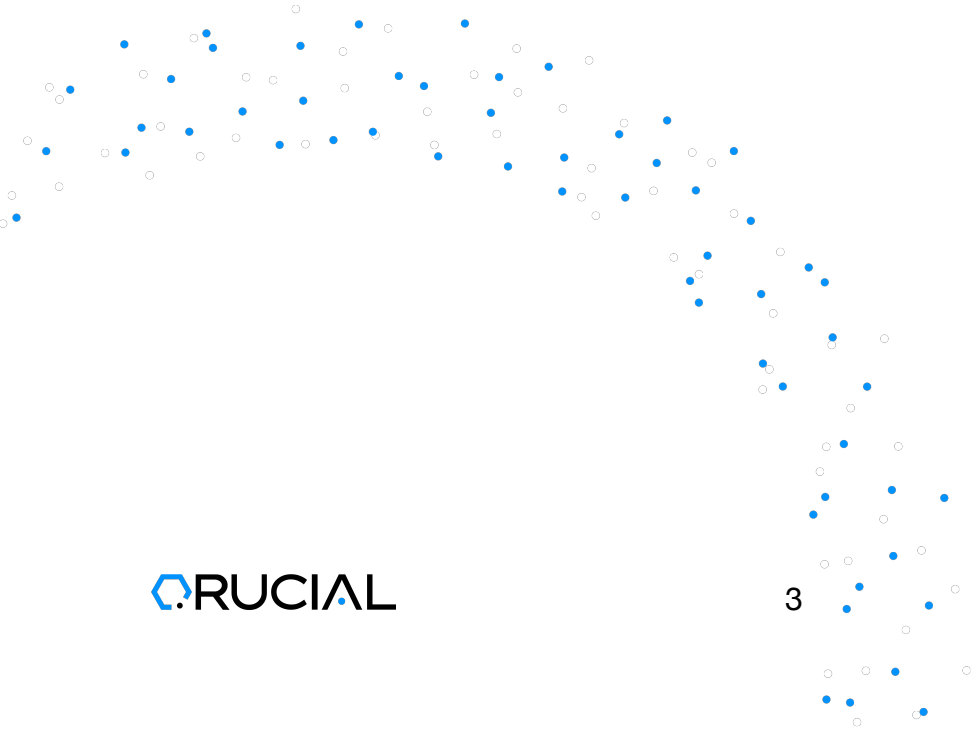
This report has been prepared for ink!. An extensive analysis has been performed by manual review, static analysis and symbolic execution.

In our Audits, we focus on the following areas:

- Analysing that the smart contract's logic meet the intentions of the project.
- Examining the smart contracts against conventional and unconventional attack vectors
- Verifying the codebase meets the current industry standard and best practices
- Cross-referencing the Project against the implementation, contract, and structure of similar Industry-leading Projects
- Elaborate line by line manual review of the entire Codebase.

The findings of the security evaluation are considered low.
We advise you to address these findings as soon as possible to assure a foremost level of security for your project and community.

- Increase good coding practices for a better structure in the source code
- Increase the number of unit test to cover all possible angles of use cases
- Add more comments per function for readability.



Overview

Project Summary

Project name	ink! ERC20 implementation by Parity
Platform	ink!
Language	Rust
Codebase	https://github.com/paritytech/ink/blob/master/examples/erc20/lib.rs
Git commit	0d2cc5b49ea40cdd2875604149d0c11767ef1236

Audit Summary

ERC20 is a common token standard and the smart contract code in scope is becoming more widely used. In order to make the ecosystem more secure, an audit has been conducted against the current implementation.

The implementation was found to be very simple and straightforward. On the smart contract level, we have found only one low severity vulnerability, but it should be noted that ink! environments are changing often, new features are being added (eg. delegatecall type of features), hence the chance for future bugs is higher than usual.

Vulnerability Summary

Level	Total	Pending	Rejected	Accepted	Partially fixed	Fixed
Critical	0	-	-	-	-	-
High	0	-	-	-	-	-
Medium	0	-	-	-	-	-
Low	1	-	-	1	-	-
Informational	0	-	-	-	-	-

Scope

Audited Code:	ink!
Blockchain Explorer Link:	https://github.com/paritytech/ink/blob/master/examples/erc20/lib.rs
Compiler:	cargo 1.62.0-nightly (dba5baf 2022-04-13)
Number of files:	1
Scope (list of files)	lib.rs

Risk Classifications

Critical:
Vulnerabilities that can lead to a loss of funds, impairment, or control over the system or its function.
We recommend that findings of this classification are fixed immediately.

High:
Findings of this classification can impact the flow of logic and can cause direct disruption in the system and the project's organization.
We recommend that issues of this classification are fixed as soon as possible.

Medium:
Vulnerabilities of this class have impact on the flow of logic, but does not cause any disturbance that would halt the system or organizational continuity.
We recommend that findings of this class are fixed nonetheless.

Low:
Bugs, or vulnerability that have minimal impact and do not pose a significant threat to the project or its users.
We recommend that issues of this class are fixed nonetheless because they increase the attack surface when your project is targeted by malicious actors.

Informational:
Findings of this class have a negligible risk factor but refer to best practices in syntax, style or general security.

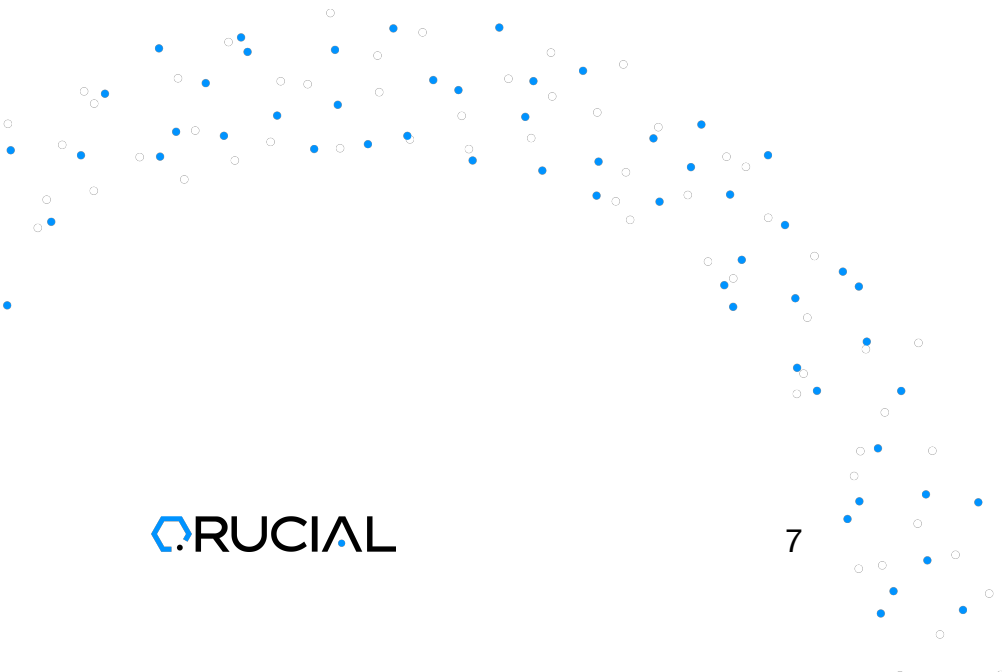
LOW: Lack of withdraw(), locking sent funds to the smart contract address

Description:
This is a common issue seen in the cryptocurrency ecosystem. When smart contracts without withdraw() function are deployed and coins/tokens are sent to them, they cannot be moved anymore, locking the funds sent.

Impact:
Users might sent coin or tokens to the contract, losing their funds.

Recommendations:
Implement the withdraw() function for coins and tokens.

References:
https://soliditydeveloper.com/eip-165



DISCLAIMER

This report is fixed to the scope and subject to terms and conditions of the service agreement provided to the customer. This report must not be referred, transmitted or disclosed to a third party without QRUCIAL's prior written consent.

This report is not an endorsement disapproval of a team, a product, a service, a company, or an individual. This report should not be considered as financial advice and does not indicate any financial or economic value in an asset, an asset class a product or service. This report is not to be seen as an indication of the legal compliance regarding of a project an asset, an asset class or a business model.

This report does not provide the guarantee, that a project is without bugs, errors vulnerabilities or code that is harmful to machines, software, or data. This report is also no indication of the validity of any business model or technology. Each individual organization is responsible to do their own due diligence or security assessment. This report is not to be seen as a guarantee of the functionality of a technology or its security. The use of access or information in this audit is used on the risk of the reader or user of this document.

This report holds no guarantee that the given information meets requirements of any kind, is compatible with applications, any software or systems. It is also not guaranteed that this audit is free of errors or harmful code or will cause interruptions of any software or systems. We do not give any guarantee of accuracy, reliability, or correctness of the information given in this audit. All third-party material provided to the client may be subject to the terms and conditions of third parties. All third-party material provided is provided without the guarantee of correctness. No third-party has the right to use the trademark QRUCIAL, its products or services as a reference or endorsement of its own products or services without prior written consent.

THANK YOU.

Our team gave their full commitment to craft this audit with precision and focus on details with the goal to support you improving your work.

With this audit we want to provide you a guidance to make your project more
secure and for presenting your
community a product they can trust in.

We make security our priority, so you don‘t have to.

RUCIAL

----Polkadot{.js} account validation address----

5EHagRYLNsCJUx5bA5Y8MZWLqPVzqQbFMC76V68Wzv7GHHXD

----Polkadot{.js} account validation address----

QRUCIAL

QRUCIAL OÜ c/o Gate to Baltics
Narva mnt 7-652, 10117 | Tallinn, Estonia