CERTIK

Security Assessment

Trustswap Inc

Jun 18th, 2021

Table of Contents

Summary

This report has been prepared for Trustswap Inc. smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

No notable vulnerabilities were identified in the codebase and it makes use of the latest security principles and style guidelines.

Overview

Project Summary

Project Name	Trustswap Inc
Description	A typical ERC-20 token factory.
Platform	Ethereum
Language	Solidity
Codebase	https://ropsten.etherscan.io/address/0xf94413cf315cb461637be61c2b9cf2a4b457a466#code
Commit	

Audit Summary

Delivery Date	Jun 18, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	ERC-20 Token, Token Factory

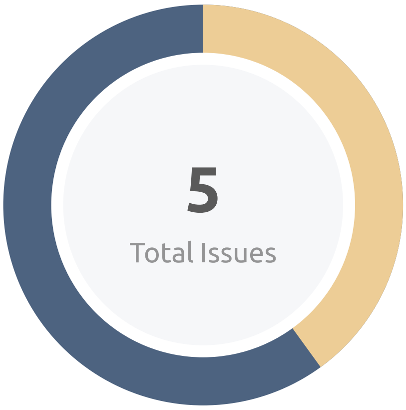
Vulnerability Summary

Total Issues	5
Critical	0
Major	0
Medium	0
Minor	2
Informational	3
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
TTT	TeamToken.sol	638d9bc7f10ca14973f2a7d33c318461c1e1d44cc7acf8a989d3209e105a7984
TTF	TeamTokenFactory.sol	3ab8b84bccc1f9790c9da7c57b1210939651f7404917af56f5e2e279cb77af80

Findings



Critical	**0** (0.00%)
Major	**0** (0.00%)
Medium	**0** (0.00%)
Minor	**2** (40.00%)
Informational	**3** (60.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
TTF-01	`event` Optimization	Language Specific	Informational	Resolved
TTF-02	Inexistent Input Sanitization	Logical Issue	Minor	Resolved
TTF-03	Return Variable Utilization	Gas Optimization	Informational	Resolved
TTF-04	Function Visibility Optimization	Gas Optimization	Informational	Resolved
TTT-01	Inexistent Input Sanitization	Logical Issue	Minor	Resolved

TTF-01 | `event` Optimization

Category	Severity	Location	Status
Language Specific	Informational	TeamTokenFactory.sol: 10	Resolved

Description

The `TeamTokenCreated` event declaration can add its `address` & `string` arguments on the `topics` data structure, hence allowing for easier off-chain monitoring.

Recommendation

We advise to use the `indexed` attribute on the `address` & `string` arguments.

Alleviation

The development team opted to consider our references and added the `indexed` attribute to the `address` & `string` arguments.

TTF-02 | Inexistent Input Sanitization

Category	Severity	Location	Status
Logical Issue	Minor	TeamTokenFactory.sol: 12~14, 16~18	Resolved

Description

The linked functions fail to check the values of the arguments.

Recommendation

We advise to add a `require` statement, checking the input values against the zero address.

Alleviation

The development team opted to consider our references and added the `checkIsAddressValid` modifier in the codebase.

TTF-03 | Return Variable Utilization

Category	Severity	Location	Status
Gas Optimization	Informational	TeamTokenFactory.sol: 20	Resolved

Description

The linked function declarations contain explicitly named `return` variables that are not utilized within the function's code block.

Recommendation

We advise that the linked variables are either utilized or omitted from the declaration.

Alleviation

The development team opted to consider our references and removed the return variable.

TTF-04 | Function Visibility Optimization

Category	Severity	Location	Status
Gas Optimization	Informational	TeamTokenFactory.sol: 20	Resolved

Description

The linked function is declared as `external` and contains array-based function arguments.

Recommendation

We advise that the array-based arguments change their data location from `memory` to `calldata`, optimizing the gas cost of the function.

Alleviation

The development team opted to consider our references and added the `calldata` attribute to the array-based function arguments.

TTT-01 | Inexistent Input Sanitization

Category	Severity	Location	Status
Logical Issue	Minor	TeamToken.sol: 8~19	Resolved

Description

The constructor fails to check the values of the arguments.

Recommendation

We advise to add `require` statements, checking that token information will not be blank and the wallet addresses will be non-zero addresses.

Alleviation

The development team opted to consider our references and added the `checkIsAddressValid` modifier in the codebase along with `require` statements.

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK's prior written consent.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.

CERTIK