CERTIK

Security Assessment

Trust Token

May 20th, 2021

Table of Contents

Summary

This report has been prepared for Trust Token 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 medium to informational. The team has alleviated most of the findings and opted to not alleviate issues that pose no threat to the system.

Overview

Project Summary

Project Name	Trust Token
Description	Trust Smart Contrats Implementation.
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/trusttoken/smart-contracts
Commits	a96a83e6fd8511f9aad748a4a5194685a27d06f3

Audit Summary

Delivery Date	May 20, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

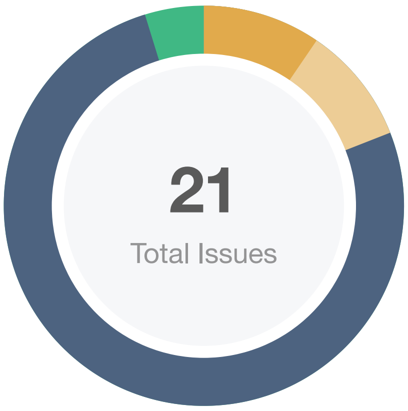
Vulnerability Summary

Total Issues	21
● Critical	0
● Major	0
● Medium	2
● Minor	2
● Informational	16
● Discussion	1

Audit Scope

ID	file	SHA256 Checksum
GAN	governance/GovernorAlpha.sol	230be4241700255aed9599a217134a8d473be8ab48c63ed3c77dda76f717390b
STT	governance/StkTruToken.sol	0f12aceaf727730dcd8ba6c6b51664b65a5a306954df33d97f11a2670ea4f65d
TIM	governance/Timelock.sol	11b32ee08dcf275e0dc5fb94d41bf12c3c99feb31f75305470062aa81d25c23e
VTN	governance/VoteToken.sol	5f12cb3d9e8b34ccbd6511eb96293c09b736356306db9dd016e8a52336743096
LIQ	truefi2/Liquidator2.sol	8a49bd39a5556f191875ac5fa3983a3c3af1738feeb556006f9deaee0181fb00
LFN	truefi2/LoanFactory2.sol	0d1f7a0118ad8d53f44b1fd8ac0fefbfd7a48ae73e4010b655f1c1bce37e6b17
LTN	truefi2/LoanToken2.sol	070b3b3ae72c3b21872fe1593423ed68e2c2138415439d326df41c63a7836fc9
PFN	truefi2/PoolFactory.sol	5aaec5ce1f473ebb79898e2a6e09cad8f8ae4ef1c0410e73caf7eefca5068a46
TFP	truefi2/TrueFiPool2.sol	a6353138aa5828affef59b6a72546232f09db982f5b63d752b6311a5023a69d6
TLN	truefi2/TrueLender2.sol	e76d83285feae6c9063fbfb62c72c07edb31fa59ada18a44bb048f9ecfaa62ae
CYS	truefi2/strategies/CurveYearnStrategy.sol	1abfc44806fdec21dd011a57aafa66327e6f6217a9d6da34759e2b4b79b3a71c

Findings



🟥 **Critical**	**0** (0.00%)
🟧 **Major**	**0** (0.00%)
🟨 **Medium**	**2** (9.52%)
🟨 **Minor**	**2** (9.52%)
🟦 **Informational**	**16** (76.19%)
🟩 **Discussion**	**1** (4.76%)

ID	Title	Category	Severity	Status
CYS-01	Visibility Specifiers Missing	Language Specific	● Informational	✓ Resolved
GAN-01	Redundant Check Against Constant Value	Language Specific, Logical Issue	● Informational	✓ Resolved
LTN-01	Constant Variable Naming	Language Specific, Coding Style	● Informational	✓ Resolved
LTN-02	Wrong Comment	Logical Issue	● Informational	✓ Resolved
LTN-03	Magic Number	Coding Style	● Informational	✓ Resolved
LTN-04	Redundant Calculation	Coding Style, Logical Issue	● Informational	ⓘ Acknowledged
PFN-01	Missing Zero Address Check	Control Flow	● Informational	✓ Resolved
STT-01	Visibility Specifiers Missing	Language Specific	● Informational	✓ Resolved
STT-02	Use of non safe transfer	Language Specific	● Minor	✓ Resolved
STT-03	Return Variable Declaration	Coding Style	● Informational	ⓘ Acknowledged
STT-04	Require With No Error Message	Language Specific	● Informational	✓ Resolved
TFP-01	Invalid Check	Language Specific	● Discussion	ⓘ Acknowledged
TFP-02	Inefficient Pattern	Logical Issue	● Minor	✓ Resolved
TFP-03	Magic Numbers	Coding Style	● Informational	✓ Resolved

ID	Title	Category	Severity	Status
TFP-04	Missing Zero Address Check	Control Flow, Language Specific	● Informational	✓ Resolved
TFP-05	Proper Representation	Logical Issue	● Informational	ⓘ Acknowledged
TFP-06	Missing Zero Address Check	Language Specific, Control Flow	● Informational	ⓘ Acknowledged
TLN-01	Missing Check	Logical Issue	● Medium	✓ Resolved
TLN-02	Code Design	Logical Issue	● Medium	ⓘ Acknowledged
TLN-03	Visibility Specifiers Missing	Language Specific	● Informational	✓ Resolved
TLN-04	Return Variable Declaration	Coding Style	● Informational	✓ Resolved

CYS-01 | Visibility Specifiers Missing

Category	Severity	Location	Status
Language Specific	● Informational	truefi2/strategies/CurveYearnStrategy.sol: 29, 31	✓ Resolved

Description

The linked variable declarations do not have a visibility specifier explicitly set.

Recommendation

Inconsistencies in the default visibility the Solidity compilers impose can cause issues in the functionality of the codebase. We advise that visibility specifiers for the linked variables are explicitly set.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

GAN-01 | Redundant Check Against Constant Value

Category	Severity	Location	Status
Language Specific, Logical Issue	● Informational	governance/GovernorAlpha.sol: 361	✓ Resolved

Description

The code contains a check expressed against a constant value. x == false while x is a bool.

Recommendation

Consider removing the constant right part.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

LTN-01 | Constant Variable Naming

Category	Severity	Location	Status
Language Specific, Coding Style	● Informational	truefi2/LoanToken2.sol: 37	✓ Resolved

Description

The linked variable does not have a proper ALL_CAPS name deviating from soliditys standards.

Recommendation

Consider renaming the variable with ALL_CAPS.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

LTN-02 | Wrong Comment

Category	Severity	Location	Status
Logical Issue	● Informational	truefi2/LoanToken2.sol: 434	✓ Resolved

Description

The function interest has commenting that does not proper represent the functionality.

Recommendation

Consider refactoring the comment.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

LTN-03 | Magic Number

Category	Severity	Location	Status
Coding Style	● Informational	truefi2/LoanToken2.sol: 268, 439	✓ Resolved

Description

The code represents constantly a value of 10000.

Recommendation

Consider adding an immutable constant.

LTN-04 | Redundant Calculation

Category	Severity	Location	Status
Coding Style, Logical Issue	● Informational	truefi2/LoanToken2.sol: 268	ⓘ Acknowledged

Description

The code redundant calculates interest variable while there is a interest function.

Recommendation

Consider using the interest function for the calculation.

Alleviation

The team has acknowledged the issue and opted to not alleviate it in the current iteration.

PFN-01 | Missing Zero Address Check

Category	Severity	Location	Status
Control Flow	● Informational	truefi2/PoolFactory.sol: 139	✓ Resolved

Description

The function setTrueLender does not check against a zero address.

Recommendation

Consider implementing a check.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

STT-01 | Visibility Specifiers Missing

Category	Severity	Location	Status
Language Specific	● Informational	governance/StkTruToken.sol: 25, 26, 58	✓ Resolved

Description

The linked variable declarations do not have a visibility specifier explicitly set.

Recommendation

Inconsistencies in the default visibility the Solidity compilers impose can cause issues in the functionality of the codebase. We advise that visibility specifiers for the linked variables are explicitly set.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

STT-02 | Use of non safe transfer

Category	Severity	Location	Status
Language Specific	● Minor	governance/StkTruToken.sol: 478	✓ Resolved

Description

The code uses transfer while SafeERC20 is available.

Recommendation

Consider using the safe functionality from the SafeERC20 library.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

STT-03 | Return Variable Declaration

Category	Severity	Location	Status
Coding Style	● Informational	governance/StkTruToken.sol: 594	ⓘ Acknowledged

Description

The linked function declarations contain explicitly named return variables that are degrading the readability of the code.

Recommendation

We advise that the linked variables are omitted from the declaration and introduced inside the functions scope.

Alleviation

The team has acknowledged the issue and opted to not alleviate it in the current iteration.

STT-04 | Require With No Error Message

Category	Severity	Location	Status
Language Specific	● Informational	governance/StkTruToken.sol: 264, 334, 478	✓ Resolved

Description

The code contains require checks with no error messages.

Recommendation

Consider adding error messages.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

TFP-01 | Invalid Check

Category	Severity	Location	Status
Language Specific	● Discussion	truefi2/TrueFiPool2.sol: 336, 378	ⓘ Acknowledged

Description

The code contains a check against the tx.origin .

Recommendation

Consider refactoring the code taking under consideration the upcoming hard fork of Ethereum and the changes on tx.origin .

TFP-02 | Inefficient Pattern

Category	Severity	Location	Status
Logical Issue	● Minor	truefi2/TrueFiPool2.sol: 557	✓ Resolved

Description

The code performs a redundant allocation to check and return the same value.

Recommendation

Consider checking the value directly returning early and allocating later if the check was successful.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

TFP-03 | Magic Numbers

Category	Severity	Location	Status
Coding Style	● Informational	truefi2/TrueFiPool2.sol: 313, 332, 382, 406, 423, 430	✓ Resolved

Description

The code represents constantly values that could be declared as constants.

Recommendation

Consider adding an immutable constant representation for those values.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

TFP-04 | Missing Zero Address Check

Category	Severity	Location	Status
Control Flow, Language Specific	● Informational	truefi2/TrueFiPool2.sol: 322	✓ Resolved

Description

The function setBeneficiary does not check against a zero address.

Recommendation

Consider implementing a check.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

TFP-05 | Proper Representation

Category	Severity	Location	Status
Logical Issue	● Informational	truefi2/TrueFiPool2.sol: 414	ⓘ Acknowledged

Description

The code represents a formula but deviates from the usage of the api.

Recommendation

Consider refactoring the code and use the provided api.

Alleviation

The team has acknowledged the issue and opted to not alleviate it in the current iteration.

TFP-06 | Missing Zero Address Check

Category	Severity	Location	Status
Language Specific, Control Flow	● Informational	truefi2/TrueFiPool2.sol: 525	ⓘ Acknowledged

Description

The function setOracle does not check against a zero address.

Recommendation

Consider implementing a check.

Alleviation

The team has acknowledged the issue and opted to not alleviate it in the current iteration.

TLN-01 | Missing Check

Category	Severity	Location	Status
Logical Issue	● Medium	truefi2/TrueLender2.sol: 189	✓ Resolved

Description

The function setLoansLimit does not check if the newLoansLimit is not bigger that the previous limit.

Recommendation

Consider implementing a check to ensure newLoansLimit > oldLoansLimit.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

TLN-02 | Code Design

Category	Severity	Location	Status
Logical Issue	● Medium	truefi2/TrueLender2.sol: 386	ⓘ Acknowledged

Description

The function distribute does not check if msg.sender is the intended one and additionally calls functionality that is commented as helper test function in L423.

Recommendation

Consider providing a rationale.

Alleviation

The team has acknowledged the issue and opted to not alleviate it in the current iteration.

TLN-03 | Visibility Specifiers Missing

Category	Severity	Location	Status
Language Specific	● Informational	truefi2/TrueLender2.sol: 41	✓ Resolved

Description

The linked variable declarations do not have a visibility specifier explicitly set.

Recommendation

Inconsistencies in the default visibility the Solidity compilers impose can cause issues in the functionality of the codebase. We advise that visibility specifiers for the linked variables are explicitly set.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

TLN-04 | Return Variable Declaration

Category	Severity	Location	Status
Coding Style	● Informational	truefi2/TrueLender2.sol: 222, 310, 339	✓ Resolved

Description

The linked function declarations contain explicitly named return variables that are degrading the readability of the code.

Recommendation

We advise that the linked variables are omitted from the declaration and introduced inside the functions scope.

Alleviation

The team has fixed the issue in commit c4fa8070358199c53e437dfc995298e1e759fff1.

Appendix

Finding Categories

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.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Language Specific

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

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

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

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