CERTIK

Preliminary Comments

Taraxa

Jul 17th, 2021

Table of Contents

Summary

This report has been prepared for Taraxa Foundation to discover issues and vulnerabilities in the source code of the Taraxa project 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.

Overview

Project Summary

Project Name	Taraxa
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/Taraxa-project/tara-erc20
Commit	73bfd3d12b9430c5e20856b4bee5187ef2a97c80

Audit Summary

Delivery Date	Jul 17, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Vulnerability Level	Total	Pending	Partially Resolved	Resolved	Acknowledged	Declined
Critical	0	0	0	0	0	0
Major	2	2	0	0	0	0
Medium	0	0	0	0	0	0
Minor	1	1	0	0	0	0
Informational	6	6	0	0	0	0
Discussion	1	1	0	0	0	0

Audit Scope

ID	file	SHA256 Checksum
STC	Staking.sol	df3a6c3b9dc194ef78e38443be9b5f273b4b6920c52e71f16b723f4d1fbe6c44
SPT	StakingProxy.sol	4b452c011411b4c6ba4e4c3fcc2455657ce90e0185d46ef8d233e60c96a294b3
SPA	StakingProxyAdmin.sol	1778a1d30b529ff7e7eb73ecbff9593ada873d9d425ba05f0d42644caaf12aa6

Understandings

Overview

This Protocol provides a staking pool that is used to lock `TARA` tokens. The rewards that the participants get for locking their tokens are calculated by an external service based on the events that this contract emits. The initial lock period is 30 days. But it can be updated later. Every time users stake tokens to the pool, the staking end time will be prolonged but the start time unchanged.

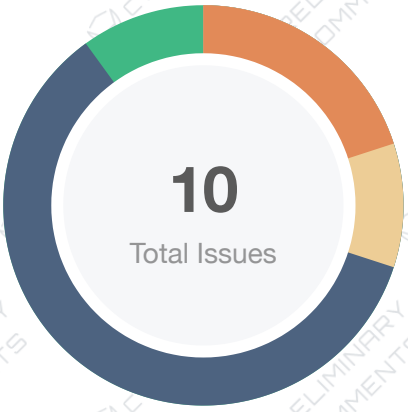
Privileged Functions

The project contains the following privileged functions that are restricted by the `owner`. This role is used to modify the contract configurations and address attributes. We grouped these functions below:

Contract `Staking`:

- function setLockingPeriod(uint256 _lockingPeriod)

Findings



Critical	**0** (0.00%)
Major	**2** (20.00%)
Medium	**0** (0.00%)
Minor	**1** (10.00%)
Informational	**6** (60.00%)
Discussion	**1** (10.00%)

ID	Title	Category	Severity	Status
SPA-01	Unlocked Compiler Version Declaration	Language Specific	Informational	Pending
SPT-01	Unlocked Compiler Version Declaration	Language Specific	Informational	Pending
STC-01	Lack of Input Validation	Logical Issue	Informational	Pending
STC-02	Missing Event Emitting	Logical Issue	Informational	Pending
STC-03	Unlocked Compiler Version Declaration	Language Specific	Informational	Pending
STC-04	Public Function That Could Be Declared External	Gas Optimization	Informational	Pending
STC-05	Check Effect Interaction Pattern Violated	Logical Issue	Minor	Pending
STC-06	Incorrect Event Data	Logical Issue	Major	Pending
STC-07	Introduction About The Staking Activity	Logical Issue	Discussion	Pending
STC-08	Privileged Ownership	**Centralization / Privilege**	**Major**	**Pending**

SPA-01 | Unlocked Compiler Version Declaration

Category	Severity	Location	Status
Language Specific	Informational	projects/Taraxa/StakingProxyAdmin.sol: 3	Pending

Description

The compiler version utilized throughout the project uses a version range, denoting that a compiler in the range can be used to compile the contracts. Recommend the compiler version should be consistent throughout the codebase.

Recommendation

It is a general practice to instead lock the compiler at a specific version rather than allow a range of compiler versions to be utilized to avoid compiler-specific bugs and be able to identify ones more easily. We recommend locking the compiler at the lowest possible version that supports all the capabilities wished by the codebase. This will ensure that the project utilizes a compiler version that has been in use for the longest time and as such is less likely to contain yet-undiscovered bugs.

SPT-01 | Unlocked Compiler Version Declaration

Category	Severity	Location	Status
Language Specific	Informational	projects/Taraxa/StakingProxy.sol: 3	Pending

Description

The compiler version utilized throughout the project uses a version range, denoting that a compiler in the range can be used to compile the contracts. Recommend the compiler version should be consistent throughout the codebase.

Recommendation

It is a general practice to instead lock the compiler at a specific version rather than allow a range of compiler versions to be utilized to avoid compiler-specific bugs and be able to identify ones more easily. We recommend locking the compiler at the lowest possible version that supports all the capabilities wished by the codebase. This will ensure that the project utilizes a compiler version that has been in use for the longest time and as such is less likely to contain yet-undiscovered bugs.

STC-01 | Lack of Input Validation

Category	Severity	Location	Status
Logical Issue	Informational	projects/Taraxa/Staking.sol: 39	Pending

Description

Addresses should be checked before assigning to make sure they are not zero addresses.

Recommendation

We recommended adding validation to check this like bellow:

```
    function initialize(address _tokenAddress) public initializer {
        require(_tokenAddress != address(0), "_tokenAddress is zero address!");
        owner = msg.sender;
        token = IERC20(_tokenAddress);
        lockingPeriod = 30 days;
    }
```

STC-02 | Missing Event Emitting

Category	Severity	Location	Status
Logical Issue	Informational	projects/Taraxa/Staking.sol: 49	Pending

Description

Functions that affect the status of sensitive variables should be able to emit events as notifications to customers.

Recommendation

We recommended adding events for sensitive actions and emit them in the function.

STC-03 | Unlocked Compiler Version Declaration

Category	Severity	Location	Status
Language Specific	Informational	projects/Taraxa/Staking.sol: 3	Pending

Description

The compiler version utilized throughout the project uses a version range, denoting that a compiler in the range can be used to compile the contracts. Recommend the compiler version should be consistent throughout the codebase.

Recommendation

It is a general practice to instead lock the compiler at a specific version rather than allow a range of compiler versions to be utilized to avoid compiler-specific bugs and be able to identify ones more easily. We recommend locking the compiler at the lowest possible version that supports all the capabilities wished by the codebase. This will ensure that the project utilizes a compiler version that has been in use for the longest time and as such is less likely to contain yet-undiscovered bugs.

STC-04 | Public Function That Could Be Declared External

Category	Severity	Location	Status
Gas Optimization	Informational	projects/Taraxa/Staking.sol: 39, 49, 71, 92	Pending

Description

public functions that are never called by the contract should be declared external to save gas.

Recommendation

Use the external attribute for functions never called from the contract.

STC-05 | Check Effect Interaction Pattern Violated

Category	Severity	Location	Status
Logical Issue	Minor	projects/Taraxa/Staking.sol: 71	Pending

Description

The order of external call/transfer and storage manipulation must follow check effect interaction pattern.

Recommendation

We advice client to check if storage manipulation is before the external call/transfer operation. For example:

```
function stake(uint256 _amount) public {
    require(_amount > 0, 'Staking: Amount cannot be zero');

    uint256 startTime = block.timestamp;
    uint256 endTime = block.timestamp.add(lockingPeriod);

    Stake storage currentStake = stakes[msg.sender];
    if (currentStake.startTime > 0) {
        currentStake.amount = currentStake.amount.add(_amount);
        currentStake.endTime = endTime;
        startTime = currentStake.startTime;
    } else {
        Stake memory newStake = Stake(_amount, startTime, endTime);
        stakes[msg.sender] = newStake;
    }

    emit Deposited(msg.sender, _amount, startTime, endTime);

    require(token.transferFrom(msg.sender, address(this), _amount));
}
```

STC-06 | Incorrect Event Data

Category	Severity	Location	Status
Logical Issue	Major	projects/Taraxa/Staking.sol: 104	Pending

Description

According to the logic of function `unstake()`, `currentStake.amount` had been cleared before emit event. Therefore, `Withdrawn` event will record wrong data.

Recommendation

Consider changing code like below:

```
function unstake() public {
    ...
    uint256 amount = currentStake.amount;
    currentStake.amount = 0;
    emit Withdrawn(msg.sender, amount);
    require(token.transfer(msg.sender, amount));
}
```

STC-07 | Introduction About The Staking Activity

Category	Severity	Location	Status
Logical Issue	Discussion	projects/Taraxa/Staking.sol: 10	Pending

Description

We noticed that in `Staking` contract, there is no reward to be distributed to users after they stake tokens to this contract. Could you please give an introduction about this staking activity? How can users get reward if they stake tokens to this contract?

STC-08 | Privileged Ownership

Category	Severity	Location	Status
Centralization / Privilege	Major	projects/Taraxa/Staking.sol: 49	Pending

Description

The owner of the contract `Staking` has the permission to change the `lockPeriod` which will affect the time users withdraw their tokens without obtaining the consensus of the community.

Recommendation

Renounce ownership when it is the right timing, or gradually migrate to a timelock plus multisig governing procedure and let the community monitor in respect of transparency considerations.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

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.