CERTIK

Security Assessment

TokensFarm

Sept 14th, 2021

Table of Contents

Summary

This report has been prepared for Tokensfarm.com to discover issues and vulnerabilities in the source code of the TokensFarm 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;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	TokensFarm
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/Tokensfarm/tokensfarm-contracts/commit/50c4350280f188649210dcd7001cb77ddd86b5f3 https://github.com/Tokensfarm/tokensfarm-contracts/commit/70c5ee4e89573912382735626a82ba00beba7e26
Commit	

Audit Summary

Delivery Date	Sept 14, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	1	0	0	0	0	1
Major	2	0	0	0	1	1
Medium	0	0	0	0	0	0
Minor	4	0	0	1	0	3
Informational	5	0	0	1	0	4
Discussion	0	0	0	0	0	0

Audit Scope

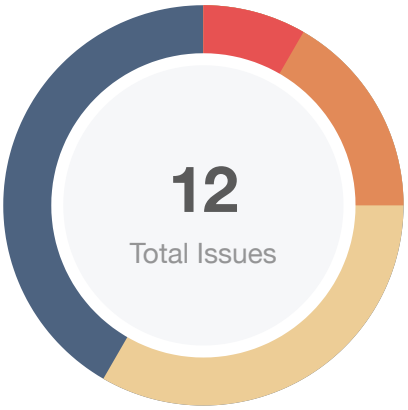
ID	File	SHA256 Checksum
TFT	TokensFarm.sol	3af98340188f09295fd7b6a097b54c894300dd9f5ee4195717c13754717b680c

It should be noted that the system design includes a number of economic arguments and assumptions. These were explored to the extent that they clarified the intention of the code base, but we did not audit the mechanism design itself.

Additionally, financial models of blockchain protocols need to be resilient to attacks. It needs to pass simulations and verifications to guarantee the security of the overall protocol. The accuracy of the financial model is not in the scope of the audit.

Please note that, according to the current logic, only the current `owner` can call the `fund()` function the second time.

Findings



Critical	**1** (8.33%)
Major	**2** (16.67%)
Medium	**0** (0.00%)
Minor	**4** (33.33%)
Informational	**5** (41.67%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
TFT-01	Set `immutable` to Variables	Gas Optimization	Informational	Acknowledged
TFT-02	Missing Emit Events	Gas Optimization	Informational	Resolved
TFT-03	Lack of Stake Validity Checks	Logical Issue	Informational	Resolved
TFT-04	Incompatibility With Deflationary Tokens	Logical Issue	Minor	Acknowledged
TFT-05	Check Effect Interaction Pattern Violated	Logical Issue	Minor	Resolved
TFT-06	Centralization Risk	**Centralization / Privilege**	**Major**	Partially Resolved
TFT-07	Logic Issue of `withdraw()`	Logical Issue	Major	Resolved
TFT-08	Lack of Input Validation	Volatile Code	Minor	Resolved
TFT-09	Lack of Error Message	Coding Style	Informational	Resolved
TFT-10	Division Before Multiplication	Language Specific	Informational	Resolved
TFT-11	Calculation Error	Logical Issue	Critical	Resolved
TFT-12	Potential Overflow	Mathematical Operations	Minor	Resolved

TFT-01 | Set `immutable` to Variables

Category	Severity	Location	Status
Gas Optimization	Informational	TokensFarm.sol: 33, 35, 37, 41, 49	Acknowledged

Description

The variables `isEarlyWithdrawAllowed`, `erc20`, `rewardPerBlock`, `startTime` and `minTimeToStake` are only changed once in the `constructor` function.

Recommendation

We advise the client to set `isEarlyWithdrawAllowed`, `erc20`, `rewardPerBlock`, `startTime` and `minTimeToStake` as `immutable` variables.

Alleviation

No alleviation.

TFT-02 | Missing Emit Events

Category	Severity	Location	Status
Gas Optimization	Informational	TokensFarm.sol: 77	Resolved

Description

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

Recommendation

We advise the client to add events for sensitive actions and emit them in the function as follows.

```
event EarlyWithdrawPenaltyChange(EarlyWithdrawPenalty penalty);

function setEarlyWithdrawPenalty(EarlyWithdrawPenalty _penalty) external onlyOwner {
    require(isEarlyWithdrawAllowed, "Early withdrawal is not allowed, so there is no
penalty.");
    penalty = _penalty;
    emit EarlyWithdrawPenaltyChange(penalty);
}
```

Alleviation

The client heeded our advice and resolved this issue in commit : efd4b84bef9eccc2a71a1415ea8389bfe8b01784.

TFT-03 | Lack of Stake Validity Checks

Category	Severity	Location	Status
Logical Issue	Informational	TokensFarm.sol: 117, 123, 209, 252, 145	Resolved

Description

There's no sanity check to validate if a stake exists.

Recommendation

We advise the client to adopt following modifier `validateStakeByStakeId` to functions `deposited()`, `pending()`, `depositTimestamp()`, `withdraw()` and `emergencyWithdraw()`.

```
modifier validateStakeByStakeId(address _user, uint256 stakeId) {
    require (stakeId < stakeInfo[_user].length , "Stake does not exist") ;
    _;
}
```

Alleviation

The client heeded our advice and resolved this issue in commit : efd4b84bef9eccc2a71a1415ea8389bfe8b01784.

TFT-04 | Incompatibility With Deflationary Tokens

Category	Severity	Location	Status
Logical Issue	Minor	TokensFarm.sol: 183, 209	Acknowledged

Description

The contract operates as the main entry for interaction with staking users. The staking users deposit LP tokens into the pool and in return get a proportionate share of the pool's rewards. Later on, the staking users can withdraw their own assets from the pool. In this procedure, `deposit()` and `withdraw()` are involved in transferring users' assets into (or out of) the protocol. When transferring standard ERC20 deflationary tokens, the input amount may not be equal to the received amount due to the charged (and burned) transaction fee. As a result, this may not meet the assumption behind these low-level asset-transferring routines and will bring unexpected balance inconsistencies.

Recommendation

We advise the client to regulate the set of LP tokens supported in the contract. If there is a need to support deflationary tokens, add necessary mitigation mechanisms to keep track of accurate balances.

Alleviation

No alleviation.

TFT-05 | Check Effect Interaction Pattern Violated

Category	Severity	Location	Status
Logical Issue	Minor	TokensFarm.sol: 252, 209	Resolved

Description

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

- `withdraw()`
- `emergencyWithdraw()`

Recommendation

We advise the client to adopt the `nonReentrant` modifier from openzeppelin library to the function `emergencyWithdraw()` and `withdraw()` to prevent any reentrancy issue or use the checks-effects-interactions pattern as follows. (LINK)

Alleviation

The client heeded our advice and resolved this issue in commit : efd4b84bef9eccc2a71a1415ea8389bfe8b01784.

TFT-06 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	**Major**	TokensFarm.sol: 77, 99	Partially Resolved

Description

To bridge the gap in trust between the `owner` and users, the `owner` needs to express a sincere attitude regarding the considerations of the administrator team's anonymity. The `owner` has the responsibility to notify users about the following capabilities:

- Set early withdrawal penalty through `setEarlyWithdrawPenalty()`
- Add a new lp to the pool through `addPool()`
- Set minimum time to stake through `setMinTimeToStake()`
- Set fee collector address through `setFeeCollector()`

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or via smart-contract-based accounts with enhanced security practices, e.g. Multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risks at the different levels in terms of the short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

The client partially resolved this issue by invoking the function `setEarlyWithdrawPenalty()` and `addPool()` in the function `constructor()`. The `owner` still has the capability to set minimum time to stake and set fee collector address.

TFT-07 | Logic Issue of `withdraw()`

Category	Severity	Location	Status
Logical Issue	Major	TokensFarm.sol: 209	Resolved

Description

According to the logic at L302, if the `penalty` equals to the `EarlyWithdrawPenalty.REDISTRIBUTE_REWARDS` and the `minimalTimeStakeRespected` is `false`, the `pendingAmount` will be used for refunding the farm through calling the `_fundInternal()` function. In the `_fundInternal` function, it reverts if the `amount` equals `0` or `endTime` is larger than the current timestamp, which would lead to the failure of `withdraw()`.

Recommendation

We advise the client to recheck the logic and take measures to prevent it from happening. We also advise the client to make more tests to ensure security.

Alleviation

The client resolved this issue in commit : 70c5ee4e89573912382735626a82ba00beba7e26.

TFT-08 | Lack of Input Validation

Category	Severity	Location	Status
Volatile Code	Minor	TokensFarm.sol: 76	Resolved

Description

The assigned value to `congressAddress` should be verified as a non-zero value to prevent being mistakenly assigned as `address(0)` in the `constructor()` function.

Recommendation

We advise the client to check that the address is not a zero in `constructor()` like as follows:

```
require(_congressAddress != address(0), "Wrong congress address.");
```

Alleviation

The client heeded our advice and resolved this issue in commit : 88c11f7f9845abc47323a06237390c76f1af9a6e.

TFT-09 | Lack of Error Message

Category	Severity	Location	Status
Coding Style	Informational	TokensFarm.sol: 246, 395	Resolved

Description

The convenience function `require` can be used to check for conditions and throw an exception if the condition is not met. If you do not provide a string argument to require, it will revert with empty error data, not even including the error selector. (LINK)

Recommendation

We advise the client to add error messages.

Alleviation

The client heeded our advice and resolved this issue in commit : 88c11f7f9845abc47323a06237390c76f1af9a6e.

TFT-10 | Division Before Multiplication

Category	Severity	Location	Status
Language Specific	Informational	TokensFarm.sol: 267, 403	Resolved

Description

Mathematical operations in the aforementioned function perform divisions before multiplications. Performing multiplication before division can sometimes avoid loss of precision.

Recommendation

We advise the client to apply multiplications before divisions if integer overflow would not happen in functions.

Alleviation

The client heeded our advice and resolved this issue in commit : 88c11f7f9845abc47323a06237390c76f1af9a6e.

TFT-11 | Calculation Error

Category	Severity	Location	Status
Logical Issue	Critical	TokensFarm.sol: 268, 404	Resolved

Description

In the function `deposit()`, the `stakeAmount` should be equal to the `_amount` minus the `feeAmount`. In the function `_erc20Transfer()`, the `rewardAmount` should be equal to the `_amount` minus the `feeAmount`.

Recommendation

We advise the client to adopt as follows:

```
268     uint256 stakeAmount = _amount.sub(feeAmount);
```

```
404     uint256 rewardAmount = _amount.sub(feeAmount);
```

Alleviation

The client heeded our advice and resolved this issue in commit : 88c11f7f9845abc47323a06237390c76f1af9a6e.

TFT-12 | Potential Overflow

Category	Severity	Location	Status
Mathematical Operations	Minor	TokensFarm.sol: 178	Resolved

Description

The max value of `uint` is 2**256-1 in `soldity`, we found that the aforemention code use `1e36` to improve the accuracy which may lead to overflow. In `MasterChef` of the `sushiswap`, `1e12` is used to do that.

Recommendation

We advise the client to use `1e12` or `1e18` to improve the accuracy if there is no special design.

Alleviation

The client heeded our advice and resolved this issue in commit : 70c5ee4e895739123827356 26a82ba00beba7e26.

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.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

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.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

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

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 you ("Customer" or 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, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

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.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

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.