FreshCut

AUDIT REPORT

Version 1.0.1

Serial No. 2022041300012017

Presented by Fairyproof

April 13, 2022

01. Introduction

This document includes the results of the audit performed by the Fairyproof team on the FreshCut project.

Audit Start Time:

March 9, 2022

Audit End Time:

March 10, 2022

Audited Source Files:

The source files audited include all the files with the extension "sol" as follows:

```
contracts/
├── FCDToken.sol
└── Vesting.sol

0 directories, 2 files
```

The goal of this audit is to review FreshCut’s solidity implementation for its token issuance function, study potential security vulnerabilities, its general design and architecture, and uncover bugs that could compromise the software in production.

We make observations on specific areas of the code that present concrete problems, as well as general observations that traverse the entire codebase horizontally, which could improve its quality as a whole.

This audit only applies to the specified code, software or any materials supplied by the FreshCut team for specified versions. Whenever the code, software, materials, settings, environment etc is changed, the comments of this audit will no longer apply.

— Disclaimer

Note that as of the date of publishing, the contents of this report reflect the current understanding of known security patterns and state of the art regarding system security. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk.

The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from off-chain sources are not extended by this review either.

Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit is recommended after the issues covered are fixed.

To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

— Methodology

The above files' code was studied in detail in order to acquire a clear impression of how the its specifications were implemented. The codebase was then subject to deep analysis and scrutiny, resulting in a series of observations. The problems and their potential solutions are discussed in this document and, whenever possible, we identify common sources for such problems and comment on them as well.

The Fairyproof auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Fairyproof to make sure we understand the size, scope, and functionality of the project's source code.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Fairyproof describe.
2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run the test cases.
   ii. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the source code to improve maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

— Structure of the document

This report contains a list of issues and comments on all the above source files. Each issue is assigned a severity level based on the potential impact of the issue and recommendations to fix it, if applicable. For ease of navigation, an index by topic and another by severity are both provided at the beginning of the report.

— Documentation

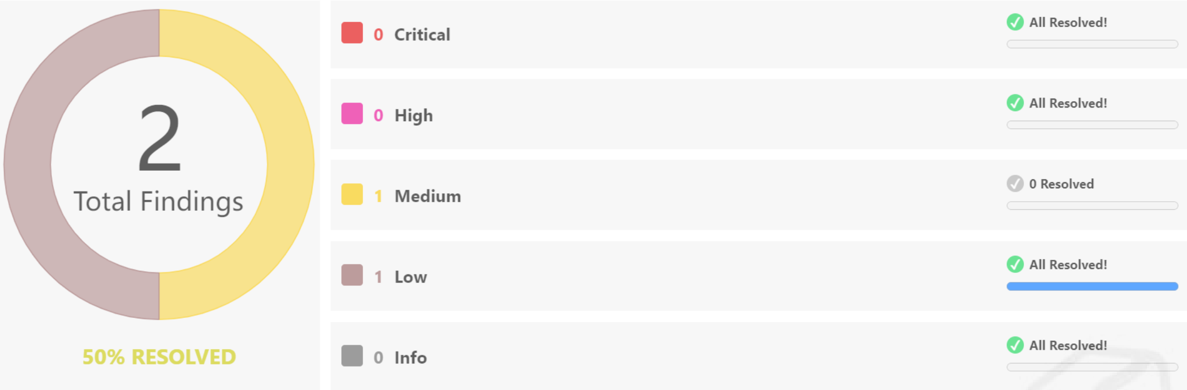
For this audit, we used the following sources of truth about how the token issuance function should work:

smart contract files

These were considered the specification, and when discrepancies arose with the actual code behavior, we consulted with the FreshCut team or reported an issue.

— Comments from Auditor

Serial Number	Auditor	Audit Time	Result
2022031000012013	Fairyproof Security Team	Mar 9, 2022 - Mar 10, 2022	Medium



Summary:

The Fairyproof security team used its auto analysis tools and manual work to audit the project. During the audit, 1 risk of medium-severity and 1 risk of low-severity were found. The risk of medium-severity has been confirmed and the risk of low-severity has been fixed.

02. About Fairyproof

Fairyproof is a leading technology firm in the blockchain industry, providing consulting and security audits for organizations. Fairyproof has developed industry security standards for designing and deploying blockchain applications.

03. Major functions of audited code

The audited code mainly implements a token issuance function and here are the details:

1. Token Issuance (FCDToken.sol)

- Token Name: FreshCut Diamond
- Token Symbol: FCD
- Token Precision: 18
- Max Supply: 1,000,000,000
- Mint/Burn: no additional minting/no token burn
- Transaction Charge: no charge of token in transfers
- Freeze/Pause Transfer: token transfer can be paused

2. Linear Vesting (Vesting.sol)

During a specified vesting period, a specified number of ERC-20 tokens and ETHs are gradually released and sent to a specified address.

3. Admin Rights

In the FCDToken.sol file, the admin can pause token transfers or pause contract upgrades.

04. Coverage of issues

The issues that the Fairyproof team covered when conducting the audit include but are not limited to the following ones:

- Re-entrancy Attack
- Replay Attack
- Reordering Attack
- Miner's Advantage
- Rollback Attack
- DDos Attack
- Transaction Ordering Attack
- Race Condition
- Access Control
- Integer Overflow/Underflow

- Timestamp Attack
- Gas Consumption
- Inappropriate Callback Function
- Function Visibility
- Implementation Vulnerability
- Uninitialized Storage Pointer
- Arithmetic Precision
- Tx.origin
- Fake Deposit
- Shadow Variable
- Design Vulnerability
- Token Issurance
- Admin Rights
- Inappropriate Proxy Design
- Inappropriate Use of Slots
- Asset Security
- Contract Upgrade/Migration
- Code Improvement
- Misc

05. Severity level reference

Every issue in this report was assigned a severity level from the following:

Critical severity issues need to be fixed as soon as possible.

High severity issues will probably bring problems and should be fixed.

Medium severity issues could potentially bring problems and should eventually be fixed.

Low severity issues are minor details and warnings that can remain unfixed but would be better fixed at some point in the future.

Informational is not an issue or risk but a suggestion for code improvement.

06. Major areas that need attention

Based on the provided souce code the Fairyproof team focused on the possible issues and risks related to the following functions or areas.

- Integer Overflow/Underflow

We checked all the code sections, which had arithmetic operations and might introduce integer overflow or underflow if no safe libraries were used. All of them used safe libraries.

We didn't find issues or risks in these functions or areas at the time of writing.

- Access Control

We checked each of the functions that could modify a state, especially those functions that could only be accessed by "owner".

We didn't find issues or risks in these functions or areas at the time of writing.

- Variable Setting

We checked whether or not the variable settings were proper.

We didn't find issues or risks in these functions or areas at the time of writing.

- State Update

We checked some key state variables which should only be set at initialization.

We didn't find issues or risks in these functions or areas at the time of writing.

- Asset Security

We checked whether or not all the functions that transfer assets were safely handled.

We didn't find issues or risks in these functions or areas at the time of writing.

- Contract Migration/Upgrade

We checked whether or not the contract files introduced issues or risks associated with contract migration/upgrade.

We found an issue, please refer to "08. Issue description" for more details.

- Functional Design

We checked whether or not the functions were designed properly.

We found an issue, please refer to "08. Issue description" for more details.

- Miscellaneous

The Fairyproof team didn't find issues or risks in other functions or areas at the time of writing.

07. List of issues by severity

Index	Title	Issue/Risk	Severity	Status
FP-1	Contract Upgradeable	Contract Upgrade/Migration	Medium	Confirmed
FP-2	Contract Receiving ETHs	Design Vulnerability	Low	✓ Fixed

08. Issue descriptions

[FP-1] Contract Upgradeable Medium Confirmed

Issue/Risk: Contract Upgrade/Migration

Description:

`FCDToken.sol` used `hardhat` to manage contract upgrades. The `owner` had the right to upgrade contracts (see the `_authorizeUpgrade` function) and the right to pause token transfers. The owner should operate this with great caution and users needed to trust the owner's operation. This was less decentralized.

Recommendation:

Consider transferring the owner's right to a multi-sig wallet or a DAO after the contract is deployed.

Update:

The FreshCut team intends for it to be that way as it wants to have the ability to upgrade the contract in the future, and to be able to pause transactions should anything nefarious arise.

Status:

It has been confirmed by the FreshCut team.

[FP-2] Contract Receiving ETHs Low ✓ Fixed

Issue/Risk: Design Vulnerability

Description:

`Vesting.sol` had redundant code including a `receive` function, therefore the contract could receive ETHs. When a user mistakenly sent ETHs to the contract, these ETHs would never be returned.

Recommendation:

Consider removing the `receive` function.

Update:

The team added a `release` function to linearly release the ETHs.

Status:

It has been fixed by the FreshCut team.

09. Recommendations to enhance the overall security

We list some recommendations in this section. They are not mandatory but will enhance the overall security of the system if they are adopted.

- N/A

Appendix:

Audited Files' SHA-256 Values:

```
FCDToken.sol: 0x86f0afd717881c46fa8b8df015c5a98b14ce29db3764788d16b7a3a512449a3e
Vesting.sol:  0xbf049abff24bbf1e8759522faf81e06dae8c29249f3dd0d0657727ff1567dd7d
```

Unit Test Result:

Compiler: `solidity: "0.8.9"`

```
> HardhatEVM: v2.8.4
> network:    hardhat



  Freshcut Diamonds contract
    Deployment
Token Supply : 1000000000000000000000000000
      ✓ Should assign the total supply of tokens to the owner
    Token Details
      ✓ has a name
      ✓ has a symbol
      ✓ has 18 decimals
    Transactions
BigNumber { value: "1000000000000000000000000000" }
      ✓ Should transfer tokens between accounts (55ms)
      ✓ Should fail if sender doesn’t have enough tokens (58ms)
    Pauseable Check
      ✓ Should pause the contract
      ✓ Should unpause the contract after its paused (43ms)
    Upgradability Testing
      ✓ Should allow to upgrade (187ms)

  VestingWallet
    Sanity
      ✓ rejects zero address for beneficiary
      ✓ check vesting contract sanity
    Deployment
      ✓ Check 0 balance release from contract (40ms)
    Vesting
      ✓ Should vest after a 1 year time (77ms)
      ✓ Should vest after full time completion (75ms)
      ✓ Should not vest after full vesting complete (74ms)
      ✓ Should not vest to anybody not assigned to  (73ms)
      ✓ Should be able to send to the vesting wallet multiple times (83ms)

  VestingWallet
    Sanity
      ✓ rejects zero address for beneficiary
      ✓ check vesting contract sanity
    Deployment
Token Supply : 1000000000000000000000000000
      ✓ Should assign the total supply of tokens to the owner
      ✓ Check funding from transfer
      ✓ Check 0 balance release from contract (46ms)
    Vesting
      ✓ Should not vest before start (65ms)
      ✓ Should vest after a 1 year time (83ms)
      ✓ Should vest after full time completion (85ms)
      ✓ Should vest full cycle with 1 year interval (162ms)
      ✓ Should not vest after full vesting complete (98ms)
      ✓ Should not vest to anybody not assigned to  (64ms)
      ✓ Should be able to send to the vesting wallet multiple times (77ms)


  29 passing (5s)
```

Coverage of Unit Test:

Note: the unit test didn't cover `FCDToken_TestContractForUpgradability.sol`

```
----------------------------------------------|----------|----------|----------|----------
|----------------|
File                                          |  % Stmts | % Branch |  % Funcs |  % Lines
|Uncovered Lines |
----------------------------------------------|----------|----------|----------|----------
|----------------|
 contracts/                                   |      100 |      100 |      100 |      100 |
                |
  FCDToken.sol                                |      100 |      100 |      100 |      100 |
                |
  Vesting.sol                                 |      100 |      100 |      100 |      100 |
                |
----------------------------------------------|----------|----------|----------|----------
|----------------|
All files                                     |      100 |      100 |      100 |      100 |
                |
----------------------------------------------|----------|----------|----------|----------
|----------------|
```

FAIRYPROOF

https://medium.com/@FairyproofT

https://twitter.com/FairyproofT

https://www.linkedin.com/company/fairyproof-tech

https://t.me/Fairyproof_tech

Reddit: https://www.reddit.com/user/FairyproofTech