Token Distributor Security Review

Pashov Audit Group

Conducted by: pashov

December 27th, 2023

Contents

1. About pashov

Krum Pashov, or **pashov**, is an independent smart contract security researcher. Having found numerous security vulnerabilities in various protocols, he does his best to contribute to the blockchain ecosystem and its protocols by putting time and effort into security research & reviews. Check his previous work here or reach out on Twitter @pashovkrum.

2. Disclaimer

A smart contract security review can never verify the complete absence of vulnerabilities. This is a time, resource and expertise bound effort where I try to find as many vulnerabilities as possible. I can not guarantee 100% security after the review or even if the review will find any problems with your smart contracts. Subsequent security reviews, bug bounty programs and on-chain monitoring are strongly recommended.

3. Introduction

A time-boxed security review of the **token-distributor** repository was done by **pashov**, with a focus on the security aspects of the application's smart contracts implementation.

4. About Token Distributor

The protocol is used to facilitate the distribution of ERC20 tokens in exchange for ETH. It is part of the **Florence Finance** ecosystem. By users contributing ETH to it, which gets directly sent to a Safe wallet that the owner of the protocol controls, they receive a token back, based on an pre-defined exchange rate.

5. Risk Classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

5.1. Impact

- High - leads to a significant material loss of assets in the protocol or significantly harms a group of users.
- Medium - only a small amount of funds can be lost (such as leakage of value) or a core functionality of the protocol is affected.
- Low - can lead to any kind of unexpected behavior with some of the protocol's functionalities that's not so critical.

5.2. Likelihood

- High - attack path is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount of funds that can be stolen or lost.
- Medium - only a conditionally incentivized attack vector, but still relatively likely.
- Low - has too many or too unlikely assumptions or requires a significant stake by the attacker with little or no incentive.

5.3. Action required for severity levels

- Critical - Must fix as soon as possible (if already deployed)
- High - Must fix (before deployment if not already deployed)
- Medium - Should fix
- Low - Could fix

6. Security Assessment Summary

review commit hash **-** **3b5d0c74c2375815f313d9a5616ecf8068739e7c**

fixes review commit hash **-** **eb02818e559bc2e20b0169497b0563e76d39c0f3**

7. Executive Summary

Over the course of the security review, pashov engaged with Florence Finance to review Token Distributor. In this period of time a total of **2** issues were uncovered.

Protocol Summary

Protocol Name	Token Distributor
Repository	token-distributor
Date	December 27th, 2023

Findings Count

Severity	**Amount**
Medium	1
Low	1
Total Findings	**2**

Summary of Findings

ID	**Title**	**Severity**	**Status**
[M-01]	DoS attack on initialization is possible	Medium	Resolved
[L-01]	Insufficient input validation	Low	Acknowledged

8. Findings

8.1. Medium Findings

[M-01] DoS attack on initialization is possible

Severity

Impact: High, as the initialization can be blocked

Likelihood: Low, as it a front-running type of an attack with no benefit for the attacker

Description

The `initializeDistributor` method in `TokenDistributor` has the following check:

```
require(token.balanceOf(address(
  token.balanceOf
)
```

This gives the expectation that the owner will pre-transfer let's say 10 tokens to the contract and then set the `_totalTokensToDistribute` argument to 10 when calling `initializeDistributor`. The problem with this is that if a malicious user front-runs the owner call with a transfer of 1 wei worth of `token` to the contract, the check and the transaction will revert as the balance will not be equal anymore.

Recommendations

Change the code in the following way:

```diff
- require(
-     token.balanceOf(address(this)) == _totalTokensToDistribute,
-     "totalTokensToDistribute must match token balance of contract"
- );
+ token.safeTransferFrom(msg.sender, address(this), _totalTokensToDistribute);
```

and do not pre-transfer tokens to the contract.

8.2. Low Findings

[L-01] Insufficient input validation

In `initializeDistributor` multiple parameters are insufficiently validated:

- `_maxEthContributionPerAddress` is only checked that is > 0, but this seems like too low of a limit , maybe use 1e18
- `_distributionStartTimestamp` is checked that it is `>= block.timestamp` but it isn't checked that it isn't too further away in the future
- `_distributionEndTimestamp` is checked that it is `>` `_distributionStartTimestamp` but it isn't checked that it isn't too further away in the future

For `_maxEthContributionPerAddress` consider using a bigger lower limit, like `1e18` for example. For `_distributionStartTimestamp` check that it isn't more than 1 week or month, or year away in the future, depending on your use case, and for `_distributionEndTimestamp` check that it isn't more than (again) a week or a month or a year away after the start timestamp.