Furucombo

COMBO & Vesting Contracts

Security Assessment

March 20th, 2021

Audited By:
Alex Papageorgiou @ CertiK
alex.papageorgiou@certik.org
Reviewed By:
Camden Smallwood @ CertiK
camden.smallwood@certik.org

Disclaimer

What is a CertiK report?

- A document describing in detail an in depth analysis of a particular piece(s) of source code provided to CertiK by a Client.
- An organized collection of testing results, analysis and inferences made about the structure, implementation and overall best practices of a particular piece of source code.
- Representation that a Client of CertiK has completed a round of auditing with the intention to increase the quality of the company/product's IT infrastructure and or source code.

Overview

Project Summary

Project Name	Furucombo - COMBO & Vesting Contracts
Description	A vesting contract and a typical ERC20 token.
Platform	Ethereum; Solidity, Yul
Codebase	0x5a7434f0579354fb51eab6f848cbda4eaa53756f 0xfFffFffF2ba8F66D4e51811C5190992176930278
Commits	N/A

Audit Summary

Delivery Date	March 20th, 2021
Method of Audit	Static Analysis, Manual Review
Consultants Engaged	1
Timeline	March 18th, 2021 - March 20th, 2021

Vulnerability Summary

Total Issues	3
🔴 **Total Critical**	0
🟠 **Total Major**	0
🟡 **Total Medium**	0
🔵 **Total Minor**	1
🟢 **Total Informational**	2

Executive Summary

We were tasked with auditing the codebase of two deployed contracts as well as a contract repository of Furucombo encompassing their COMBO token, rCOMBO token meant to represent an IOU and finally a token vesting contract.

We were not able to pinpoint any severe vulnerabilities to the system, however, we did detect certain points where better security practices can be applied as well as a single point where the design can be optimized better towards the ideals of the project.

All outward and inward transfers of the system conform to the Checks-Effects-Interactions pattern and no common vulnerabilities such as re-entrancies were identified.

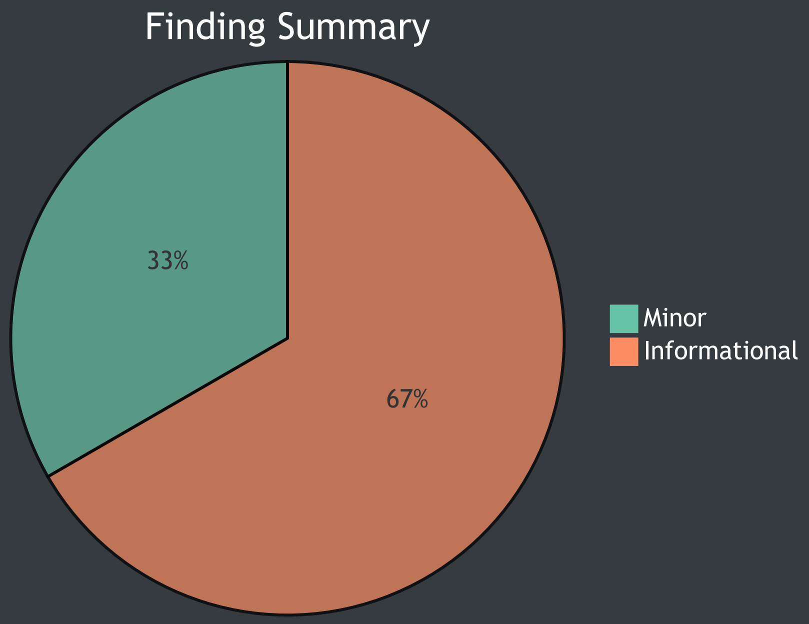
System Analysis

The vesting system (`TokenVesting.sol`) is controlled by the `owner` who is also able to revoke certain vests at will without affecting the already awarded balance however. As such, we believe it to be sufficiently fair in its operation.

Files In Scope

ID	Contract	Location
COM	COMBO.sol	COMBO.sol
TVG	TokenVesting.sol	TokenVesting.sol

File Dependency Graph



Review Findings

ID	Title	Type	Severity	Resolved
COM-01	Redundant Getter Invocation	Gas Optimization	🟢 Informational	ⓘ
TVG-01	Inexistance of Pull-Over-Push Pattern	Standard Conformity	🔵 Minor	ⓘ
TVG-02	Variable Mutability Specifier	Gas Optimization	🟢 Informational	ⓘ

COM-01: Redundant Getter Invocation

Type	Severity	Location
Gas Optimization	● Informational	COMBO.sol L737

Description:

The `constructor` of the `COMBO` token utilizes the `decimals` getter variable redundantly so as the `decimals` is equal to `18` when not manually set within the OpenZeppelin library.

Recommendation:

We advise it to be removed from the codebase and swapped by the `18` value literal.

Alleviation:

The Furucombo team has stated that this finding doesn't affect the functionality of the contract and as such, will not be updated to the live deployment of `COMBO` .

TVG-01: Inexistance of Pull-Over-Push Pattern

Type	Severity	Location
Language Specific	● Minor	TokenVesting.sol L92-L96

Description:

The transfer of ownership in the `Ownable` implementation is not conforming to the pull-over-push pattern and directly overwrites the previous `owner`.

Recommendation:

We advise the pull-over-push pattern to be applied whereby a new owner is first proposed and consequently needs to accept ownership to prove that the address can actuate transactions.

Alleviation:

The Furucombo development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase due to time constraints.

TVG-02: Variable Mutability Specifier

Type	Severity	Location
Gas Optimization	● Informational	TokenVesting.sol L577, L608

Description:

The linked variable declaration is only assigned to once during the contract's `constructor` .

Recommendation:

We advise the `immutable` trait to be introduced to the linked declaration to greatly optimize the gas cost involved in utilizing it.

Alleviation:

The Furucombo development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase due to time constraints.

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings refer to exhibits that 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.

Language Specific

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