byterocket

Smart Contract Audit Report

Castello

Token Contract

7th of January 2022

Contents

1. Preface

The developers of the **Castello** Token Contract contracted byterocket to conduct a smart contract audit of their token contract. The contract features an upgradeable ERC20 contract for mainnet, together with an upgradeable Polygon version of their token that directly integrates with the official bridge between Polygon and Ethereum.

Castello aims to "*revolutionize the cryptocurrency market and combine the worlds of artistry and blockchain technology via a blockchain, digitizing the funding process for a world-class artwork as well as its future use cases*".

The team of byterocket reviewed and audited the above smart contracts in the course of this audit. We started on the 3rd and finished on the 7th of January 2022.

The audit included the following services:

- *Manual Multi-Pass Code Review*
- *Protocol/Logic Analysis*
- *Automated Code Review*
- *Formal Report*

byterocket gained access to the code via their private Github Repository. We based the audit on the master branch's state on December 1st, 2021 (*commit hash 30dd2466dec57bc506ba590717e3804b32405371*).

2. Manual Code Review

We conducted a manual multi-pass code review of the smart contracts mentioned in section (1). Three different people went through the smart contract independently and compared their results in multiple concluding discussions.

These contracts are written according to the latest standards used within the Ethereum community and the Solidity community's best practices. The naming of variables is very logical and understandable, which results in the contract being useful to understand. The code is very well documented and up to the latest standards.

The developers included all of the relevant scripts and tests, which allowed us to run through all of the processes.

On the code level, we **found no bugs or flaws**. A further check with multiple automated reviewing tools (*MythX*, *Slither*, *Manticore*, *and different fuzzing tools*) **did not find any additional bugs**.

3. Protocol/Logic Review

Part of our audits are also analyses of the protocol and its logic. A team of three auditors went through the implementation and documentation of the implemented protocol.

We went through all of the provided documentation, tests, and contracts in a very detailed manner. The general description of the protocol is very well made, it's very easy to understand how each function is supposed to work and what it implements.

We were **not able to discover any problems** in the protocol implemented in the smart contract.

3.1 Token Implementation [Token.sol]

This contract implements an ERC20 token, based on the upgradeable standard by OpenZeppelin. The contract as well as the included AccessControl package is properly initialized. The deployer will become the admin of the contracts as well as receive the initial supply of the token. The token has 8 decimals.

The developers also included a recommended 50-slot storage gap for future upgrades to the storage layout.

3.2 Polygon Implementation [TokenPolygon.sol]

The polygon version of the token properly implements the required functionalities which are needed to work with the official PoS bridge between Ethereum and Polygon. The contract from (3.1) is imported and only extended by the two functions `deposit()` and `withdraw()`. The `deposit()` function can only be called by the "`child chain manager`" (which is correctly set to "0xA6FA4fB5f76172d178d61B04b0ecd319C5d 1C0aa"), in case a user has deposited tokens on the Ethereum mainnet with the intent to transfer them to the Polygon chain. The `withdraw()` function can be called by any token holder on Polygon intending to transfer their tokens back to Ethereum mainnet.
These corresponding functions are minting or burning the tokens on the Polygon chain.

Again, the developers included a recommended 50-slot storage gap for future upgrades to the storage layout.

4. Summary

During our code review (*which was done manually and automated*), we **found no bugs or flaws**. Our automated systems and review tools also **did not find any additional ones**.

The protocol review and analysis did neither uncover any game-theoretical nature problems nor any other functions prone to abuse.

During our multiple deployments to various local testnets, we haven't been able to find any problems or unforeseen issues.

In general, we are **delighted** with the overall quality of the code and its documentation.