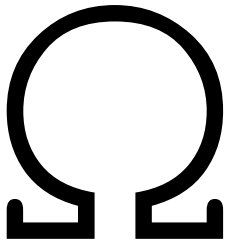
Backed ERC20
Contract Changes

Audit Report

December 28, 2022



Team Omega

Teamomega.eth.limo

Summary

Backed Finance has asked Team Omega to audit the contracts that define the behavior of their ERC20 tokens.

Omega has audited the Backed Finance ERC20 Token in May 2022. That report can be found here: ipfs://QmWmj4gMNWaCM2n5xwq4D44Wo3R6c9yTaVear3K3ZpokhW.

The present report covers changes that were made after that date. These changes concern the integration of a blacklist contract by chainAnalysis.

We found **no high severity issues** - these are issues that can lead to a loss of funds, and are essential to fix. We classified **no** issues as “medium” - these are issues we believe you should definitely address. We did classify **1** issue as “low”, and an additional issue was classified as “info” - we believe the code would improve if these issues were addressed as well.

Subsequently, Backed Finance addressed all of the issues we found, and no open issues remain.

Severity	**Number of issues**	**Number of resolved issues**
High	0	0
Medium	0	0
Low	1	1
Info	1	1

Scope of the Audit

The audit concerns files developed in the following repository:

`https://github.com/backed-fi/backed-token-contract/`

Specifically, it concerns any changes made since our last report from May of this year, i.e. the difference between:

`15dce14123fd7358c907710c5592a22443d3e87d`

and commit

`327a4b8d9156b6a664bfd10ca58142ab83704ad2`

These changes add a blacklist functionality to the contracts

Resolution

After receiving a preliminary report, Backed Finance has addressed the issues mentioned in this report in the following commit:

`40a1c0584d3f1e82e24f98f1eb00575ec7403f31`

The changes were checked by Omega, and we have added a description of the resolution to each of the issues

Methods Used

Code Review

We manually inspected the source code to identify potential security flaws. The contracts were compiled, deployed, and tested in a test environment.

Automatic analysis

We have used static analysis tools to detect common potential vulnerabilities. No serious issues were identified with the automated processes.

Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

Severity definitions

High	Vulnerabilities that can lead to loss of assets or data manipulations.
Medium	Vulnerabilities that are essential to fix, but that do not lead to assets loss or data manipulations
Low	Issues that do not represent direct exploit, such as poor implementations, deviations from best practice, high gas costs, etc
Info	Matters of opinion

Findings

G1. Blacklister can disable mint and burn [low] [resolved]

The blacklist address can blacklist the 0 address `address(0)` . Because minting (and burning) is represented internally in the ERC20 contracts as receiving from (or sending to) the zero address, this effectively disables minting and burning to any address.
This seems to give the blacklist more power than intended
Recommendation: Do not allow to add the 0 address to the blacklist
Resolution: The issue was resolved as recommended

G2. Blacklist does not extend to spender in transferFrom [info] [resolved]

A blacklisted address cannot receive or send tokens. A blacklisted address can, however, be approved to send tokens for other accounts. Other ERC20 tokens with a blacklist functionality (like the USDC contract) also extend You should consider extending the blacklist functionality also the `spender` role in the `transferFrom` implementation.
Recommendation: Forbid a blacklisted address from sending tokens for others using `transferFrom`
Resolution: The issue was resolved as recommended