LKD Token audit, October 8th, 2018

LakeDiamond Token

Audit report

Overview

The audited software consists of the following solidity files:

- LKD.sol

Each section will detail the audit of each individual file, with a final conclusion and list of recommendations at the end of this document.

LKD.sol

- Deployment cost with solc 0.4.25: 1754986 gas
- Deployment cost with solc 0.4.25 if optimized: 846708 gas
- Total supply: 141,120,000
- Decimals: 0
- Ticker: LKD

Overview

The Token contract is based on the standard burnable token library by OpenZeppelin V2, and as such all functionality inherited from that library is considered stable.

The token defines itself as a token of LKD ticker symbol with a total supply of 141,120,000 tokens, 0 decimals each. Each token represents a minute of time, and this unit of time is indivisible in the system it's going to be running on. The token can be burned only by the holder of the tokens, and only if the holder has the tokens in their address. The owner cannot burn someone else's tokens.

Machine analysis report

Machine analysis via Securify, Mythril and Manticore has found no issues.

Serious issues

- None

Medium issues

- None

Aesthetic or trivial issues

- The library the token is based on is in beta. While the newer version is much more robust and structured than v1, and was developed with readability and improved security in mind, the fact that it is beta means that some bugs might be present in

the library as a whole. However, it is the opinion of this team that no such danger exists for the token itself as it is inheriting from straighforward, easy to read, easy to understand, well tested code that has been used thousands of times in thousands of projects with marketcaps hundreds of factors higher than the one in this project, without a single recorded incident.

Conclusion

The token contract contains no vulnerabilities that investors or developers should be worried about. It can be considered ready to deploy with a final readiness rating of A.

General Recommendations

- We recommend using the optimizer with 200 runs to reduce deployment cost by around 50%
- We recommend adding token metadata to the token's contract on Etherscan as soon as possible to give it a distinct look and additional information
- We recommend validating the contract code on Etherscan through the verification UI
- We recommend approaching popular wallets for the reason of adding the token into their lists. One such early-adopter wallet is the Status IM client, but others apply as well.
- We recommend purchasing an ENS domain and resolving the token's address to something like token.lakediamond.eth for increased readability and reliability