Overview

DATA v2 Smart Contract Audit Report

Prepared for Streamr Network AG by Isentropy LLC

Auditor: Jonathan Wolff jwolff@isentropy.com

June 29 2021

Overview

Project Name: DATA v2

Code: https://github.com/streamr-dev/DATAv2

Commit: 6af417bc86cbc27b996372160e26b4ec5afec8da

Files:

- contracts/DATAv2.sol
- contracts/DataTokenMigrator.sol
- contracts/IERC677.sol
- contracts/IERC677Receiver.sol

Description: DATA v2 is an upgrade to the existing DATA ERC20 token that provides the following features:

- “permit” functionality that allows approval of token transfers by submission of a signature.
- “burn” function that allows token holders to burn tokens in a way that can easily be detected by event analysis.
- ERC677 transferAndCall functon that notifies contract token transfer recipients via a callback function.
- “mint” function that allows specified Minters to mint coins at will. Streamr has indicated that they will manage minting of the DATA v2 coin.

The code is substantially built from OpenZeppelin’s open-source contract library, with a small number of custom additions. OpenZeppelin’s code is out of the scope of this audit.

Methodology:

1. Discuss design with the Streamr team.
2. Read Solidity code and tests, compile and test DATA v2 with the Hardhat setup provided. Check for attack vectors (re-entrancy, access control, DoS, replay, overflow, etc), gas usage, interface.
3. Document issues along with suggested improvements.

Findings

Critical Issues:
None found

Major Issues:
None found

Warnings:
The following are not problems by themselves, but could become problems through malicious behavior or error:

WARN1: DATAv2.sol has admin role and possibly multiple minter roles, who can mint tokens at will. DATA v2 is not a “trustless” token. This is by design. The security of DATA v2 is dependent on external actors.

DATA v2 uses OpenZeppelin’s AccessControl framework to configure roles. In the default setup, there is a master Admin role that can add and remove Minters at will.

Recommendation:
1. Document Streamr’s policy to DATA v2 admin and minting in README.
2. Configure the contract in a way that limits that ability of malicious actors to control Admin and Minter roles, eg using multisig wallets.

WARN2: There is no limit to how much a Minter can mint.

One malicious or erroneous mint operation could have catastrophic consequences for the whole token ecosystem. It seems sensible to have some limit here to reduce that risk.

Recommendation: Implement per-Minter minting limit.

Findings

Minor Issues:

MINOR1: transferAndCall does not check the return value of onTokenTransfer from the recipient contract. This behavior might be unintuitive and should be clarified.

The ERC677 standard is somewhat vague about what should occur if onTokenTransfer is unimplemented or returns success = false in the recipient contract. Major ERC677 projects such as LINK also do not check the return value of onTokenTransfer.

Recommendation: Describe the behavior of transferAndCall with respect to onTokenTransfer in the README. Add unit tests to show desired behavior. For example add unit test to show that transferAndCall reverts if onTokenTransfer unimplemented in recipient contract.

MINOR2: Users and other smart contracts cannot easily see the list of Minters from the DATA v2 contract.

Minters are stored in a Solidity mapping, which cannot be iterated over to produce a list. This could be an interface hurdle for users who wish to vet the Minters.

Recommendation: Streamr should assure that users have some way to reliably list the Minters. The list of Minters can be collected by a blockchain indexer like The Graph.

Other Comments:

We noticed that permit functionality is based on a "draft" contract from OpenZeppelin. The name implies it might not be production ready, but we did not notice any problem in particular. OpenZeppelin is not in the scope of this audit. While we could not find a major project that uses this exact code, DAI, UNI and GRT's permit mechanisms are quite similar. The signed message format used by permit is based on EIP712.