1 Introduction

This documentation briefly describes the submitted BDA Project 2 solution. The author is Vojtěch Fiala (xfiala61). The solution allows the user to interact with an improved ERC20 token. Many of the functions are based on a consensus – the users with admin roles have to manually vote for each proposal they want to pass, unless they proposed it – then they voted automatically.

2 Features & Disadvantages

The solution uses a *MetaMask*[1] address to interact. To do that, the user has to have the MetaMask extension downloaded and must have imported one of the valid private keys from Ganache.

Two of these keys are presented in README.md

The solution is based on three contracts.

- ImprovedERC – The main contract which inherits from others.
- MintManager – ImprovedERC inherits from this. The point of the contract is to manage everything to do with Minting – mintAdmins, TMAX, one-time going over the TMAX value...
- TransferManager – ImprovedERC inherits from this. The contract manages everything to do with transfers – restrAdmins, TRANSFERLIMITs for each user...

The contracts use mappings to save each proposal. This proposal contains information about what is being proposed. However, no voting result is being saved. This means that it's very hard to find whether the user has already voted. The user's vote is recorded for each proposal he voted for, but for example finding all proposals the user voted in would be very difficult. Despite this, the voting is prevented from replayAttacks – each user can only vote once.

This solution does not support delegation.

[1] `https://metamask.io/`

3 Other technical limitations

Other than the disadvantages on backend, there are other technical limitations such as:

- The application does not support changing the active wallet account - in such a case you need to restart the application (F5)
- Application does not work on Mozzila Firefox browser, the only supported browser is Google Chrome
- For the application to work, the following must be true for erc_config.js - TOTALSUPPLY $\geq$ TMAX $\geq$ TRANSFERLIMIT
- Updating the frontend only happens arbitrarily when a new block is added to the chain. The latest data can always be retrieved after a refresh (F5)

4 Gas consumption

The estimated gas usage of all tested operations was at most in several hundred thousand per operation. Gas information was found from the given tests file.

- Mint – minting is one of the more expensive operations, costs over 190 000 units of gas
- Sending – Sending tokens is less expensive than minting, costs only about 60 000-75 000 units of gas
- Proposals – Proposals are somehow expensive, usually costing more than 150 000 units of gas. Even 200 000 was measured.
- Voting – Voting usually costs less, only about 100 000 units of gas.