Mainframe Secondary Audit

This smart contract audit was prepared by Quantstamp, the protocol for securing smart contracts.

This security audit report follows a generic template. Future Quantstamp reports will follow a similar template and they will be fully generated by automated tools.

Specification

Our understanding of the specification was based on the following documentation:

- Mainframe Token Contracts README

We also reviewed all instructions provided in the Github repository, ERC20 branch during the time of audit.

Methodology

The review was conducted during 2018-July-04 by the Quantstamp team, which included auditing intern Nadir Akhtar and senior engineer Martin Derka.

Their procedure can be summarized as follows:

1. Code review
    1. Review of the specification
    2. Manual review of code
    3. Comparison to specification
2. Itemize recommendations

Source Code

The following source code was reviewed during the audit.

Repository	Commit
contracts (branch: ERC20)	162b46a

Security Audit

This Mainframe Token Security Audit is to provide the Mainframe team with a cursory look into their ERC20 fork of their `contracts` GitHub repository. The report aims to identify any issues or vulnerabilities arising from the transition from an ERC827 token standard to an ERC20 token standard. Because of limited time to perform this audit, only the contracts `MainframeToken.sol` and `MainframeTokenDistribution.sol` were reviewed, as they are the only ones substantially affected by the transition. This is intended to be supplementary to the previous audit, meaning that unresolved vulnerabilities from the previous audit may not be noted in this report.

Context

The `MainframeToken.sol` contract transitioned from ERC827 to ERC20. Only the new token and its new structure's implications on the rest of the contracts are in scope for this review.

Evaluation

The modified token contracts are secure. Only one method was found to be at risk of failing, but as it is callable by the owner only, it poses no danger to other users.

Method with Unlimited Gas Consumption

In smart contracts, `for` loops are often prone to vulnerabilities given the nature of transactions and the concept of gas: The gas necessary for executing such a loop is proportional to the number of iterations. Using is safe when the number of iterations is predictable, but against the best practices if unknown.

In `MainframeTokenDistribution.sol`, the `distributeTokens()` function loops through all `recipients` passed to the smart contract. Because the number of recipients is unbounded, it is possible that the transaction will consume so much gas that it will not be able to fit within a block. In addition, if the `transferFrom()` function ever failed, it would revert all progress thus far. To mitigate both these issues, it is much safer to break that functionality into smaller, digestible pieces.

Recommendation

The Quantstamp team recommends replacing the loop with method with signature `sendTokens(address tokenOwner, address recipient, uint256 value)` that implements a single iteration of that loop, and calling this method for every recipient instead. To ensure no recipient receives tokens twice, the contract can be maintain a mapping from `address` to `bool` tracking which recipients have received tokens, or require that `mainframeToken.balanceOf(recipient) == 0)`.

Other Issues

These other issues, though not immediate security vulnerabilities, were still concerns to the Quantstamp team. If possible, take some time to fix the following potential problems.

- In file `MainframeToken.sol` in the `validDestination` modifier, we suggest that `require(address != 0x0)` be included in this modifier as well.
- Both `MainframeToken#47,51` and `MainframeTokenDistribution#8,19` accept `uint` parameters. We suggest that those be turned into `uint256`.
- Consider making the `emergencyERC20Drain()` functions of `MainframeToken.sol` identical to that of `MainframeTokenDistribution.sol` to drain any and all accidental tokens, as there is likely no scenario in which it would be preferable to only drain, say, half.

Appendix

File Signatures

Below are SHA256 file signatures of the relevant files reviewed in the audit.

```
$ shasum -a 256 ./contracts/*
240b17a721e3d9e301f90a860c49b9988f8ad27a003de2a5d129c9d6a3d0f048  ./contracts/MainframeStake
dffa4902acdaa5e2fd6a5da539162680b4ae320a0d364c3e5ba870dc9f36dd49  ./contracts/MainframeToken
4318e60152f648e4fef2b55813e138cccfe96ef533bbab2b12b00ef60fd4038f  ./contracts/MainframeToken
138b99e76c412e6a7e5533c968aa07a7a7a69d43b93d83bf4062ffe9f380a5a4  ./contracts/Migrations.sol
83133eb4afed383dd8dcb4ed29e1b998c93a8a40fec88f8afa3e0ebc74f5bbbd  ./contracts/StakeInterface
```

Disclosure

Purpose of report

The scope of our review is limited to a review of Solidity code and only the source code we note as being within the scope of our review within this report. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The Solidity language itself remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond Solidity that could present security risks.

The report is not an endorsement or indictment of any particular project or team, and the report does not guarantee the security of any particular project. This report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset.

No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. Specifically, for the avoidance of doubt, this report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp Technologies Inc. (QTI). Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that QTI are not responsible for the content or operation of such web sites, and that QTI shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that QTI endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. QTI assumes no responsibility for the use of third-party software on the website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by QTI; however, QTI does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.