eRLC iExec

Consensys Diligence

Date	January 2021
Lead Auditor	Shayan Eskandari
Co-auditors	Nicholas Ward

1 Executive Summary

This report presents the results of our engagement with **iExec** to review the **eRLC** code base for KYC-enabled ERC-20 (ERC-677) tokens as well as the integration of these tokens with the iExec PoCo smart contracts.

The review was conducted by **Shayan Eskandari** and **Nicholas Ward** over the course of 10 person-days between **January 4th** and **January 8th, 2021**.

2 Scope

Our review focused on the commit hash `b16266d4940f9cc695859a47c483485c48fbda66` for eRLC and the KYC additions to the PoCo delegate modules at commit hash `96a39c9d53668896321556d23351a4e79d4d46a8`. Notably, the scope excluded the PoCo delegate mechanism itself and all other functionalities within the PoCo system. The complete list of files in scope for each repository can be found in the Appendix.

2.1 Objectives

Together with the **iExec** team, we identified the following priorities for our review:

1. Ensure that the system is implemented consistently with the intended functionality, and without unintended edge cases.
2. Identify known vulnerabilities particular to smart contract systems, as outlined in our Smart Contract Best Practices, and the Smart Contract Weakness Classification Registry.
3. Assess the design and implementation of the eRLC token contract, including KYC access controls and deposit and withdrawal functionality.
4. Review the addition of KYC-aware transfer hooks and authorization checks to the PoCo system.

3 Security Specification

This section describes, **from a security perspective**, the expected behavior of the system under review. It is not a substitute for documentation. The purpose of this section is to identify specific security properties that were validated by the audit team.

3.1 Actors

The relevant actors are listed below with their respective abilities:

eRLC Actors:

- **Admin** (`DEFAULT_ADMIN_ROLE`):
  - Manage KYC Admins
  - Initiate Snapshots of user balances `snapshot`
  - Claim any tokens sent to the eRLC contract (will be added to the admin's balance)
  - Recover unintentional deposits of the underlying token (will be added to the admin's balance)
- **KYC Admin** (`KYC_ADMIN_ROLE`): kyc admin, manages kyc members
  - Grant and revoke KYC role to a set of addresses `grantKYC` , `revokeKYC`
- **KYC Member** (`KYC_MEMBER_ROLE`):
  - Deposit, Withdraw, Receive, and Transfer tokens.
- Anyone else:
  - Due to the verification of KYC in the ERC-677 `_beforeTokenTransfer()` callback, no account without the `KYC_MEMBER_ROLE` can perform any actions in the contract.

While it is an explicitly desired property of the system, it is important to note the ability of the **KYC Admin** to revoke KYC at any time, effectively freezing the funds for a given address.

3.2 Security Properties

The following is a non-exhaustive list of security properties that were verified in this audit:

- eRLC as described in the Executive summary is mainly a KYC token, hence trusting the admins to act according to the regulations is a feature of the system. This includes but not limited to locking accounts (none-KYC members) from using the system, minting new tokens and claiming the extra tokens in the contract.
- Only the **KYC Admin** can grant or revoke KYC approval
- No address without the `KYC_MEMBER_ROLE` can deposit, transfer, or withdraw eRLC tokens

- Tokens can be minted only on deposit of an equivalent amount of the underlying asset or by the **Admin** when claiming the excess balance of the eRLC contract via `recover()`
- No amount of the underlying asset can be withdrawn from the eRLC contract without burning an equivalent amount of the eRLC token

4 Recommendations

4.1 Share status codes between `ERC20KYC` and `IexecERC20CoreKYC`

Both `ERC20KYC` and the PoCo system's `IexecERC20CoreKYC` utilize the same status codes for `ERC20KYC.detectTransferRestriction`, with a status code of 0 indicating no restriction. To prevent future changes to these status codes from raising conflicts between the two definitions, it would be beneficial to define them in a single independent library or contract, perhaps as a solidity `enum`.

code/eRLC/contracts/ERC20KYC.sol:L28-L30

```
uint8 internal constant _RESTRICTION_OK               = uint8(0);
uint8 internal constant _RESTRICTION_MISSING_KYC_FROM = uint8(0x01);
uint8 internal constant _RESTRICTION_MISSING_KYC_TO   = uint8(0x02);
```

code/PoCo/contracts/modules/delegates/IexecERC20CoreKYC.sol:L36-L40

```
uint8 restrictionCode = m_baseToken.detectTransferRestriction(from, to, amount);
if (restrictionCode != uint8(0))
{
        revert(m_baseToken.messageForTransferRestriction(restrictionCode));
}
```

4.2 eRLC: Include `Transfer(address,address,uint256,bytes)` event Acknowledged

The ERC-677 standard includes an event, `Transfer(address,address,uint256,bytes)`, to be emitted from the `transferAndCall()` function. Including this event would prevent deviation from the token standard and make it easier to trace external calls from the token contract.

However, we believe ERC-677 to be under-specified in its current form, suggesting that deviation from the standard may not warrant a change if the event is currently excluded for other reasons. Simply making this deviation explicit in user-facing documentation may suffice.

4.3 eRLC: Require a delay period before granting `KYC_ADMIN_ROLE` Acknowledged

Resolution

The development team already has plans in place to use a `TimelockController` as the `KYC_DEFAULT_ADMIN` of the eRLC contract. We agree that this is a sufficient solution and has the additional benefit of avoiding unnecessary code complexity in the eRLC contract.

The **KYC Admin** has the ability to freeze the funds of any user at any time by revoking the `KYC_MEMBER_ROLE`. The trust requirements from users can be decreased slightly by implementing a delay on granting this ability to new addresses.

While the management of private keys and admin access is outside the scope of this review, the addition of a time delay can also help protect the development team and the system itself in the event of private key compromise.

Examples

Batch granting and revoking of the `KYC_MEMBER_ROLE`. These functions can only be called by the `KYC_ADMIN_ROLE`:

code/eRLC/contracts/KYC.sol:L56-L72

```
function grantKYC(address[] calldata accounts)
external virtual override
{
    for (uint256 i = 0; i < accounts.length; ++i)
    {
        grantRole(KYC_MEMBER_ROLE, accounts[i]);
    }
}

function revokeKYC(address[] calldata accounts)
external virtual override
{
    for (uint256 i = 0; i < accounts.length; ++i)
    {
        revokeRole(KYC_MEMBER_ROLE, accounts[i]);
    }
}
```

Appendix 1 - Files in Scope

This review covered the following files:

eRLC Repository:

File	SHA-1 hash
ERC677.sol	9a187e76516e352fec834f2f77612b717e6d7bd1

File	SHA-1 hash
KYC.sol	e922130045a37dcd7ce791ac3566a08f6e3d1fbe
ERLC.sol	da987bd06bda51a2b8a2b76d6e505a9ba61c25a7
ERLCTokenSwap.sol	20ad99f44374a4e9ed43e8cd245a83201455b5f2
ERC20KYC.sol	48b443f2127724e50b6c93e67f2b5c108a113ce2
Claimable.sol	cf0ab5b6f255aa38a669032f791c6ed3089ca971
ERC20Softcap.sol	d81ba880a2879360356f31ac2f117c6bdca6ea42
interfaces/IKYC.sol	f8b8601a1bef3f8ee15b8697284e97c93397138e
interfaces/IERC20KYC.sol	9c2f0d2348444de354f5490ab47b4eec9e89904f
interfaces/IERC677Receiver.sol	55b4232894ca3cabb81433bf192da40fbeed54cd
interfaces/IERC677.sol	410c708e946bdaa845c8d263c2d5dbbeaf75bc86
interfaces/IERC1404.sol	dad28efcd76127a3bc6a9cbcd74038e44a680ebc

PoCo Repository:

File	SHA-1 hash
modules/delegates/IexecERC20CoreKYC.sol	6d516c02f71c1ff38970b8e246d133486a21804f
modules/delegates/IexecERC20DelegateKYC.sol	7b946cecff33f5bee80715103bb9f1def3c6f5c4
modules/delegates/IexecPoco1DelegateKYC.sol	fa9e7ba731f11a5a58dcdf48a9a581f529ef1b72
modules/delegates/IexecEscrowTokenDelegateKYC.sol	e221d94d3b248cc36c333da7a6c8c4b8a2015d41
modules/delegates/IexecPoco2KYCDelegate.sol	904a694194f47e43c3695949c45ae19c6bb82384

Appendix 2 - Disclosure

ConsenSys Diligence (“CD”) typically receives compensation from one or more clients (the “Clients”) for performing the analysis contained in these reports (the “Reports”). The Reports may be distributed through other means, including via ConsenSys publications and other distributions.

The Reports are not an endorsement or indictment of any particular project or team, and the Reports do 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. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any Third-Party in any respect, including regarding the bugfree nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. 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. CD owes no duty to any Third-Party by virtue of publishing these Reports.

PURPOSE OF REPORTS The Reports and the analysis described therein are created solely for Clients and published with their consent. The scope of our review is limited to a review of Solidity code and only the Solidity code we note as being within the scope of our review within this report. 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. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty.

CD makes the Reports available to parties other than the Clients (i.e., “third parties”) – on its website. CD hopes that by making these analyses publicly available, it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.

LINKS TO OTHER WEB SITES FROM THIS WEB SITE You may, through hypertext or other computer links, gain access to web sites operated by persons other than ConsenSys and CD. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites’ owners. You agree that ConsenSys and CD are not responsible for the content or operation of such Web sites, and that ConsenSys and CD 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 ConsenSys and CD 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 Reports. ConsenSys and CD assumes no responsibility for the use of third party software on the Web Site 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 Reports is current as of the date appearing on the Report and is subject to change without notice. Unless indicated otherwise, by ConsenSys and CD.