Contract Audit Results

Prepared on: Sep 22, 2020

Prepared by:
Charles Holtzkampf
Sentnl

Prepared for:
Jae Chung
Cryptolocaly

Table of Contents

Executive Summary

This document outlines any issues found during the audit of the contracts:

- GIVtoken
- Localeos

The contract has 3 flaws or security vulnerabilities. The risk associated with this contract is Low

REMARK	MINOR	MAJOR	CRITICAL
6	3	0	0

Severity Description

REMARK

Remarks are instances in the code that are worthy of attention, but in no way represent a security flaw in the code. These issues might cause problems with the user experience, confusion with new developers working on the project, or other inconveniences.

Things that would fall under remarks would include:

- Instances where best practices are not followed
- Spelling and grammar mistakes
- Inconsistencies in the code styling and structure

MINOR

Issues of Minor severity can cause problems in the code, but would not cause the code to crash unexpectedly or for funds to be lost. It might cause results that would be unexpected by users, or minor disruptions in operations. Minor problems are prone to become major problems if not addressed appropriately.

Things that would fall under minor would include:

- Logic flaws (excluding those that cause crashes or loss of funds)
- Code duplication
- Ambiguous code

MAJOR

Issues of major security can cause the code to crash unexpectedly, or lead to deadlock situations.

Things that would fall under major would include:

- Logic flaws that cause crashes
- Timeout exceptions
- Incorrect ABI file generation
- Unrestricted resource usage (for example, users can lock all RAM on contract)

CRITICAL

Critical issues cause a loss of funds or severely impact contract usage.

Things that would fall under critical would include:

- Missing checks for authorization
- Logic flaws that cause loss of funds
- Logic flaws that impact economics of system
- All known exploits (for example, on_notification fake transfer exploit)

Methodology

Throughout the review process, we check that the token contract:

- Documentation and code comments match logic and behaviour
- Is not affected by any known vulnerabilities

Our team follows best practices and industry-standard techniques to verify the proper implementation of the smart contract. Our smart contract developers reviewed the contract line by line, documenting any issues as they were discovered.

Our strategies consist largely of manual collaboration between multiple team members at each stage of the review, including:

I. Due diligence in assessing the overall code quality of the codebase.
II. Testing contract logic against common and uncommon attack vectors.
III. Thorough, manual review of the codebase, line-by-line.

Our testing includes

- Overflow Audit
- Authority Control Audit Authority Vulnerability Audit
- Authority Excessive Audit
- Safety Design Audit Hard-coded Audit
- Show coding Audit
- Abnormal check Audit
- Type safety Audit
- Denial of Service Audit
- Performance Optimization Audit
- Design Logic Audit
- False Notice Audit
- False Error Notification Audit

- Counterfeit Token Audit
- Random Number Security Audit
- Rollback Attack Audit

Structural Analysis

The Cryptolocaly smart contracts both consisted of 2 unique files. The code is split up into several functions and actions, with the functions performing work behind the scenes, and the actions being the interface to the users and to the smart contract itself.

Audit Results - GIVtoken

REMARK

Outdated ERC20 version

The ERC20 isn't the latest version.

REMARK

Outdated version of solidity

The contract is not being compiled with the latest version of solidity.

Audit Results - Localeos

REMARK

opentrade:13:

This code doesn't need to be hardcoded and can be generalised so the owner can add further tokens in the future. As it stands, adding a new token will require updating the contract.

REMARK

EOSIO_DE..:771

It's no longer required to define this macro, and it's best practice to use the eosio transfer tag as below. An asterisk for the contract name variable can be used instead to listen to all transfers.

```
[[eosio::on_notify("contractName::transfer")]] void transfer(eosio::name const &from, eosio::name const &to, eosio::asset const &quantity, std::string const &memo);
```

REMARK

refund:269:

This code doesn't need to be hardcoded and can be generalised so the owner can add further tokens in the future. As it stands, adding a new token will require updating the contract.

REMARK

send:546:

Dust will accumulate in the contract. Consider keeping track of it somewhere.

Audit Results - Localeos

MINOR

erasetrade:644

Possible a trade could be erased that contains funds. Consider adding a check to ensure the trade is empty.

MINOR

referclaim::

Possible funds can be removed but a trade for those funds still exists.

MINOR

The contract seems coupled with an off chain counterpart that looks to be managing the program logic. This potentially opens the contract up to external security flaws..

Contract Files

localeos.cpp	SHA256
GIVToken.sol	fcfdecd46c4c0b0888b6c2b1a537defb0d403113a0f91ec39c30ce5da8f1c083
localeos.cpp	7eca052a910aac32a3165af0922e68b69c39a94e5f077971d98600d49c42298b
localeos.hpp	8375ee26b35c4176ce012a7bb209bd53b2bc33d6805d7083c7f5aa82f2ecf312