COINMERCENARY / SMART CONTRACT AUDIT

Digitize Coin (DTZ)

20 MARCH 2018 / TABLE OF CONTENTS

INTRODUCTION

CoinMercenary provides comprehensive, independent smart contract auditing.

We help stakeholders confirm the quality and security of their smart contracts using our comprehensive and standardized audit process. Each audit is unbiased and verified by multiple reputable auditors.

The scope of this audit was to analyze and document the Digitize (DTZ) token generation event contract.

This audit provides practical assurance of the logic and implementation of the contract.

AUDIT METHODOLOGY

CoinMercenary audits consist of four categories of analysis.

Design Patterns

We first inspect the overall structure of the smart contract, including both manual and automated analysis.

The design pattern analysis checks appropriate test coverage, utilizes a linter to ensure consistent style and composition, and code comments are reviewed. Overall architecture and safe usage of third party smart contracts are checked to ensure the contract is structured in a way that will not result in future issues.

Static Analysis

The static analysis portion of our audit is performed using a series of automated tools, purposefully designed to test the security of the contract. These tools include:

- **Manticore** - Dynamic binary analysis tool with EVM support.
- **Mythril** - Reversing and bug hunting framework for the Ethereum blockchain.
- **Oyente** - Analyzes Solidity code to find common vulnerabilities.
- **Solgraph** - DOT graph creation for visualizing function control flow of a Solidity contract to highlight potential security vulnerabilities.

Data flow and control flow are also analyzed to identify vulnerabilities.

Manual Analysis

Performing a hands on review of the smart contract to identify common vulnerabilities is the most intensive portion of our audit. Checks for race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks are part of our standardized process.

Network Behavior

In addition to our design pattern check, we also specifically look at network behavior. We model how the smart contract will operate once in production, then determine the answers to questions such as: how much gas will be used, are there any optimizations, how will the contract interact?

Contracts Reviewed

On March 20, 2018 using git hash d5c0a0b692cb2f5de4e5f0f2eef38de144ba6943, the following contract files and their respective SHA256 fingerprints were reviewed:

Filename	**SHA256 Fingerprint**
DigitizeCoin.sol	6d63c6079a6e43bd362a3165f4cfdcfda0e6af874b20f6acc25e91c02ad979b1
DigitizeCoinPresale.sol	456c56be419828cce3a68903fc31599bfd9615cf9f2e3c59ab8ff01815d88c88
RefundVault.sol	30d3f05c9a85cfd4c64b3fbf4a64025c365f2a46de9986e05815b615762126ba
TokenVesting.sol	2fbd05e691c39ebe6376f06040c46d9bc7c709cc121839144b229b8a40c04b82

Remediation Audit

Not required.

Filename	**SHA256 Fingerprint**

AUDIT SUMMARY

The contracts have been found to be free of security issues.

Analysis Results

	Initial Audit	**Remediation Audit**
Design Patterns	Passed	Not required
Static Analysis	Updates Recommended	Not required
Manual Analysis	Passed	Not required
Token Allocation	Passed	Not required
Network Behavior	Passed	Not required

Test Results

Extensive test coverage available.

Token Allocation Results

Vesting contract available, allocation not defined in contract.

Explicit Vulnerability Check Results

Known Vulnerability	**Results**
Parity Multisig Bug 2	Not vulnerable
Callstack Depth Attack	Not vulnerable
Transaction-Ordering Dependence	Not vulnerable
Timestamp Dependency	Not vulnerable
Re-Entrancy Vulnerability	Not vulnerable

ISSUES DISCOVERED

Issues below are listed from most critical to least critical. Severity is determined by an assessment of the risk of exploitation or otherwise unsafe behavior.

Severity Levels

- **Informational** - No impact on the contract.
- **Low** - Minimal impact on operational ability.
- **Medium** - Affects the ability of the contract to operate.
- **High** - Affects the ability of the contract to work as designed in a significant way.
- **Critical** - Funds may be allocated incorrectly, lost or otherwise result in a significant loss.

Issues

DTZ-1 / Informational: Redundant fallback function

Present in DigitizeCoin.sol, lines: L76

Explanation

The payment rejection fallback is redundant. Starting from Solidity 0.4.0, contracts without a fallback function automatically revert payments, making the code above redundant.

Resolution

Not yet resolved.

CONCLUSION

The reviewed smart contract is well crafted and follows common security practices. No critical problems have been found.

The care and attention to detail by the Digitize team for the token generation event shows their commitment to security. We’re proud to have Digitize as a customer, and look forward to seeing their upcoming success.