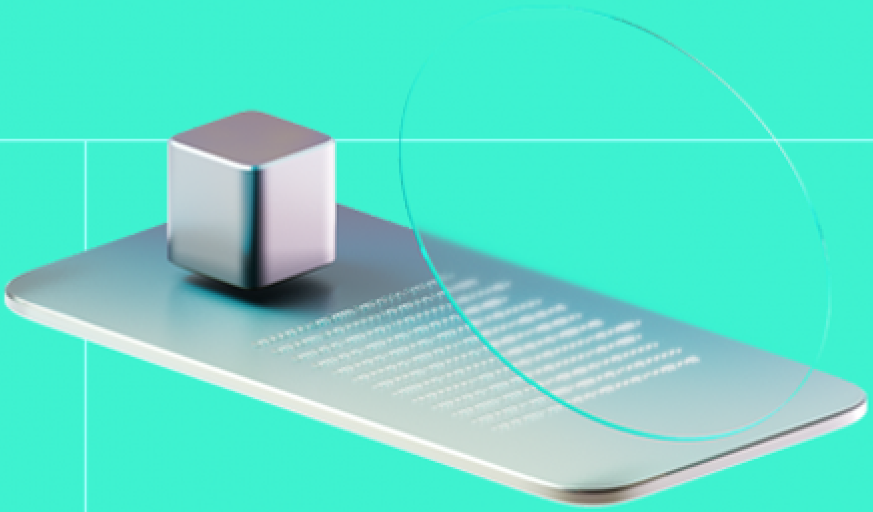
Smart Contract Code Review And Security Analysis Report



Customer: Aqueductfinance

Date: 26/07/2024

We express our gratitude to the Aqueductfinance team for the collaborative engagement that enabled the execution of this Smart Contract Security Assessment.

Aqueduct is a platform for trading OTC without an escrow.

Document

Name	Smart Contract Code Review and Security Analysis Report for Aqueductfinance
Audited By	Kornel Światłowski, Viktor Raboshchuk
Approved By	Przemyslaw Swiatowiec
Website	https://aqueduct.finance/
Changelog	24/07/2024 - Preliminary Report; 26/07/2024 - Final Report
Platform	Ethereum, Base, Polygon
Language	Solidity
Tags	Verifier, ZoneContract
Methodology	https://hackenio.cc/sc_methodology

Review Scope

Repository	https://github.com/aqueduct-finance/otc-v1
Commit	eb8a9a9cd3a904a0e6a2ffbc8d8aa9adf0909e7e

Audit Summary

The system users should acknowledge all the risks summed up in the risks section of the report

0	0	0	0
Total Findings	Resolved	Accepted	Mitigated

Findings by Severity

Severity	Count
Critical	0
High	0
Medium	0
Low	0

Documentation quality

- Functional requirements are detailed.
- Technical description has some gaps.

Code quality

- The development environment is configured.

Test coverage

Code coverage of the project is **70.83%** (branch coverage).

- Deployment and basic user interactions are covered with tests.
- Negative cases coverage is missed.
- Some functionalities are not covered with tests.

Table of Contents

System Overview **6**

Privileged Roles 6

Risks **7**

Findings **8**

Vulnerability Details 8

Observation Details 8

Disclaimers 12

Appendix 1. Severity Definitions **13**

Appendix 2. Scope **14**

System Overview

The **TokenLockupPlansVerifier** contract is a zone contract for OpenSea's Seaport protocol, enabling verification of locked token amounts in Hedgey's TokenLockupPlans contract. It ensures users can safely trade TokenLockupPlans lockups by invalidating trades if tokens are redeemed from the lockup. The contract maintains a whitelist of approved lockup addresses and verifies lockup ownership and token amounts during order validation. This validation process involves checking the provided data against the expected owner, token ID, and lockup amount.

Privileged roles

- The contract does not have privileged roles.

Risks

- **System Reliance on External Contracts**: The functioning of the system significantly relies on specific external contracts. Any flaws or vulnerabilities in these contracts adversely affect the audited project, potentially leading to security breaches or loss of funds.
- **Scope Definition and Security Guarantees:** The audit does not cover all code in the repository. Contracts outside the audit scope may introduce vulnerabilities, potentially impacting the overall security due to the interconnected nature of smart contracts.

Findings

Vulnerability Details

Observation Details

F-2024-4353 - Incorrect Seaport Protocol Version Reference in README.md - Info

Description:

According to the documentation (`README.md`), the contracts are built on top of OpenSea's Seaport protocol (version 1.5). The documentation also contains a link to the Seaport documentation for further reference. However, the provided link is incorrect because it leads to the newest version of the Seaport protocol, which is currently at version 1.6. There are significant differences between version 1.5 and 1.6, and the verified contract will not work with version 1.6. This incorrect link can lead to misunderstandings and incorrect assumptions related to the wrong Seaport version. Therefore, it is crucial to have the proper link to the correct version in the documentation.

```
# Aqueduct V1 OTC

Contracts built on top of OpenSea's seaport protocol (version 1.5). They add ext

Seaport docs: [https://docs.opensea.io/docs/seaport](https://github.com/ProjectO
```

Assets:

- contracts/zones/TokenLockupPlansVerifier.sol [https://github.com/aqueduct-finance/otc-v1]

Status:

Fixed

Recommendations

Remediation:

It is recommended to update the `README.md` file to ensure that the link directs to the correct version of the Seaport protocol documentation (version 1.5).

Resolution: The issue was resolved in commit **fa91240**. The `README.md` file has been updated with the correct version of the Seaport protocol documentation (version 1.5).

F-2024-4354 - Floating Pragma - Info

Description:

The project uses floating pragma `^0.8.20`.

A "floating pragma" in Solidity refers to the practice of using a pragma statement that does not specify a fixed compiler version but instead allows the contract to be compiled with any compatible compiler version. This issue arises when pragma statements like `pragma solidity ^0.8.20;` are used without a specific version number, allowing the contract to be compiled with the latest available compiler version. This can lead to various compatibility and stability issues.

Assets:

- contracts/zones/TokenLockupPlansVerifier.sol [https://github.com/aqueduct-finance/otc-v1]

Status: Fixed

Recommendations

Remediation: Consider locking the pragma version whenever possible and avoid using a floating pragma in the final deployment. Consider known bugs for the compiler version that is chosen.

Resolution: The issue was resolved in commit **fbad530**. The pragma has been locked to `0.8.20`.

F-2024-4356 - Missing Validation in TokenLockupPlansVerifier Constructor - Info

Description:

The constructor() of the TokenLockupPlansVerifier contract lacks any form of validation for its input parameters:

- Missing check against zero length of the provided _whitelistedLockupAddresses array. There is a possibility to deploy the contract with 0 whitelisted addresses, rendering the contract unusable.
- Missing check against the zero address (0x0) address in the provided _whitelistedLockupAddresses array. The absence of zero address control can lead to unintended behavior when a Solidity smart contract does not properly check or prevent interactions with the zero address.

```
constructor(address[] memory _whitelistedLockupAddresses) {
    for (uint256 i = 0; i < _whitelistedLockupAddresses.length; i++) {
        whitelistedLockupAddresses[_whitelistedLockupAddresses[i]] = true;
    }
}
```

Assets:

- contracts/zones/TokenLockupPlansVerifier.sol [https://github.com/aqueduct-finance/otc-v1]

Status: Fixed

Recommendations

Remediation:

It is recommended to implement validation checks in the constructor() of the TokenLockupPlansVerifier contract to ensure the _whitelistedLockupAddresses array is not empty and does not contain the 0x0 address. This will prevent deployment with invalid parameters and ensure proper contract functionality.

Resolution:

The issue was resolved in commit **1efc736**. Validation of the _whitelistedLockupAddresses array argument against the 0x0 addresses have been added to the constructor(). Similarly, the length of the _whitelistedLockupAddresses parameter is now verified.

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed based on best industry practices at the time of the writing of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The report contains no statements or warranties on the identification of all vulnerabilities and security of the code. The report covers the code submitted and reviewed, so it may not be relevant after any modifications. Do not consider this report as a final and sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements.

While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

English is the original language of the report. The Consultant is not responsible for the correctness of the translated versions.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the Consultant cannot guarantee the explicit security of the audited smart contracts.

Appendix 1. Severity Definitions

When auditing smart contracts, Hacken is using a risk-based approach that considers **Likelihood**, **Impact**, **Exploitability** and **Complexity** metrics to evaluate findings and score severities.

Reference on how risk scoring is done is available through the repository in our Github organization:

hknio/severity-formula

Severity	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to the loss of user funds or contract state manipulation.
High	High vulnerabilities are usually harder to exploit, requiring specific conditions, or have a more limited scope, but can still lead to the loss of user funds or contract state manipulation.
Medium	Medium vulnerabilities are usually limited to state manipulations and, in most cases, cannot lead to asset loss. Contradictions and requirements violations. Major deviations from best practices are also in this category.
Low	Major deviations from best practices or major Gas inefficiency. These issues will not have a significant impact on code execution, do not affect security score but can affect code quality score.

Appendix 2. Scope

The scope of the project includes the following smart contracts from the provided repository:

Scope Details	
Repository	https://github.com/aqueduct-finance/otc-v1
Commit	eb8a9a9cd3a904a0e6a2ffbc8d8aa9adf0909e7e
Whitepaper	https://docs.aqueduct.finance/docs
Requirements	https://github.com/aqueduct-finance/otc-v1/blob/main/README.md
Technical Requirements	https://github.com/aqueduct-finance/otc-v1/blob/main/README.md

Contracts in Scope
contracts/zones/TokenLockupPlansVerifier.sol