Security Assessment

O3 Swap

May 12th, 2021

Summary

This report has been prepared for O3 Swap smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Additionally, this audit is based on a premise that all external smart contracts are implemented safely.

The security assessment resulted in 9 findings that ranged from minor to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	O3 Swap
Description	Cross-chain Aggregation Protocol
Platform	Ethereum, BSC, Heco
Language	Solidity
Codebase	https://github.com/O3Labs/o3swap-aggregator-contracts/tree/c46ed522534fdfc279344a4945e9159241f2c9bf
Commits	c46ed522534fdfc279344a4945e9159241f2c9bf

Audit Summary

Delivery Date	May 12, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

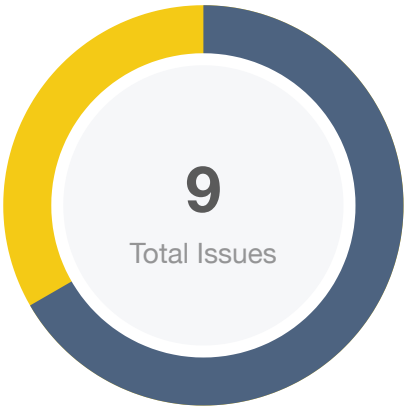
Vulnerability Summary

Total Issues	9
Critical	0
Major	0
Medium	0
Minor	3
Informational	6
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
OSB	contracts/O3SwapBSCPancakeBridge.sol	a75269106db938595c58f75279025a25c62c9b1c6262c96b4a44f88e788e4135
OSE	contracts/O3SwapETHUniswapBridge.sol	7fa8f272b8a11addec03b0c952d565e7cf7c50e85082127c1253aea56c62b344
OSH	contracts/O3SwapHecoMdexBridge.sol	37fd7b8b45e3e565c60713d3df5092a24a328a1d28b7e562cebc1e225e9587be
PLO	contracts/bsc/libraries/PancakeLibrary.sol	25ceeae402b6bc700c7751ee1f7c529442578aaa1c8a24b6a8df038c461407a9
UVL	contracts/eth/libraries/UniswapV2Library.sol	81f8defbb1ffa2fc0b05d1d5086e3fdea362de55f79e43f178020cce3aea5613
MLO	contracts/heco/libraries/MdexLibrary.sol	96b2871ca15a5dee1ac6479ff719123f56e92def4c8ae033997e3aa1a5caa928
COL	contracts/libraries/Convert.sol	a0331bbf4fb9468c5da41dfc32d1c5fdcb196d16cdb36ca26c6fabc0f08a16a1
SMO	contracts/libraries/SafeMath.sol	0c9bf711853a2ea53b393c891c8331f362ad4f66ba08e12d1b019f61db55f351
THO	contracts/libraries/TransferHelper.sol	c4d4c7416347c6fdccfdbef64f65139cab7f6f4abe2a93b258f6c0e907cb2508
MOL	contracts/utils/Migrations.sol	a5e7c17c07549e59ccd406771ba54ec2ab25ba73e1ab37bb090979678aa8e8e7
OOL	contracts/utils/Ownable.sol	3ead9dc71efb05ef89e293f74907c3117415b412d8c7fc67fafb9f415cf278fc

Findings



Critical	0 (0.00%)
Major	0 (0.00%)
Medium	0 (0.00%)
Minor	3 (33.33%)
Informational	6 (66.67%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
OSB-01	Missing Emit Events	Coding Style	Informational	Declined
OSB-02	Missing Zero Address Validation	Logical Issue	Informational	Partially Resolved
OSB-03	Unimplemented Method	Logical Issue	Minor	Acknowledged
OSE-01	Missing Emit Events	Coding Style	Informational	Declined
OSE-02	Missing Zero Address Validation	Logical Issue	Informational	Partially Resolved
OSE-03	Unimplemented Method	Logical Issue	Minor	Acknowledged
OSH-01	Missing Emit Events	Coding Style	Informational	Declined
OSH-02	Missing Zero Address Validation	Logical Issue	Informational	Partially Resolved
OSH-03	Unimplemented Method	Logical Issue	Minor	Acknowledged

OSB-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	contracts/O3SwapBSCPancakeBridge.sol: 260, 272, 276, 280, 284, 288	Declined

Description

Some functions should be able to emit events as notifications to customers because they change the status of sensitive variables or call important processes. This suggestion is not limited to these codes but also applies to other similar codes.

Recommendation

Consider adding an emit after changing the status of variables or calling important processes.

Alleviation

[O3 Swap] response: If the preset parameters are incorrect, these methods will be used for error correction, if the parameters are correct, these methods will not be used.

OSB-02 | Missing Zero Address Validation

Category	Severity	Location	Status
Logical Issue	Informational	contracts/O3SwapBSCPancakeBridge.sol: 42~44, 280, 284, 288	Partially Resolved

Description

Addresses should be checked before assignment to make sure they are not zero addresses.

Recommendation

Consider adding a zero check.

Alleviation

The team heeded our advice and partially added a zero check. Code change was applied in commit : ee0b0c395ce165c163449d5db36f83d673a5ef08.

OSB-03 | Unimplemented Method

Category	Severity	Location	Status
Logical Issue	Minor	contracts/O3SwapBSCPancakeBridge.sol: 241~251	Acknowledged

Description

The function `ISwapper(polySwapper).swap()` hasn't been implemented yet. This protocol can only be used after such functions to be implemented safely.

Alleviation

[O3 Swap] response: This is a contract developed by our partner, and it is not open source yet. We transfer tokens to a contract and lock these tokens. Then it will continue to execute the asset cross-chain contract process. We will use the audit report of the corresponding contract to show safety in the future.

OSE-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	contracts/O3SwapETHUniswapBridge.sol: 260, 272, 276, 280, 284, 288, 272, 276, 280, 284, 288	Declined

Description

Some functions should be able to emit events as notifications to customers because they change the status of sensitive variables or call important processes. This suggestion is not limited to these codes but also applies to other similar codes.

Recommendation

Consider adding an emit after changing the status of variables or calling important processes.

Alleviation

[O3 Swap] response: If the preset parameters are incorrect, these methods will be used for error correction, if the parameters are correct, these methods will not be used.

OSE-02 | Missing Zero Address Validation

Category	Severity	Location	Status
Logical Issue	Informational	contracts/O3SwapETHUniswapBridge.sol: 42~44, 280, 284, 288	Partially Resolved

Description

Addresses should be checked before assignment to make sure they are not zero addresses.

Recommendation

Consider adding a zero check.

Alleviation

The team heeded our advice and partially added a zero check. Code change was applied in commit : ee0b0c395ce165c163449d5db36f83d673a5ef08.

OSE-03 | Unimplemented Method

Category	Severity	Location	Status
Logical Issue	Minor	contracts/O3SwapETHUniswapBridge.sol: 241~251	Acknowledged

Description

The function `ISwapper(polySwapper).swap()` hasn't been implemented yet. This protocol can only be used after such functions to be implemented safely.

Alleviation

[O3 Swap] response: This is a contract developed by our partner, and it is not open source yet. We transfer tokens to a contract and lock these tokens. Then it will continue to execute the asset cross-chain contract process. We will use the audit report of the corresponding contract to show safety in the future.

OSH-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	contracts/O3SwapHecoMdexBridge.sol: 260	Declined

Description

Some functions should be able to emit events as notifications to customers because they change the status of sensitive variables or call important processes. This suggestion is not limited to these codes but also applies to other similar codes.

Recommendation

Consider adding an emit after changing the status of variables or calling important processes.

Alleviation

[O3 Swap] response: If the preset parameters are incorrect, these methods will be used for error correction, if the parameters are correct, these methods will not be used.

OSH-02 | Missing Zero Address Validation

Category	Severity	Location	Status
Logical Issue	Informational	contracts/O3SwapHecoMdexBridge.sol: 42, 280, 284, 288	Partially Resolved

Description

Addresses should be checked before assignment to make sure they are not zero addresses.

Recommendation

Consider adding a zero check.

Alleviation

The team heeded our advice and partially added a zero check. Code change was applied in commit : ee0b0c395ce165c163449d5db36f83d673a5ef08.

OSH-03 | Unimplemented Method

Category	Severity	Location	Status
Logical Issue	Minor	contracts/O3SwapHecoMdexBridge.sol: 241~251	Acknowledged

Description

The function `ISwapper(polySwapper).swap()` hasn't been implemented yet. This protocol can only be used after such functions to be implemented safely.

Alleviation

[O3 Swap] response: This is a contract developed by our partner, and it is not open source yet. We transfer tokens to a contract and lock these tokens. Then it will continue to execute the asset cross-chain contract process. We will use the audit report of the corresponding contract to show safety in the future.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Data Flow

Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a struct assignment operation affecting an in-memory struct rather than an in-storage one.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers

Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error

Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content string of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK's prior written consent.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.