Smart Contract Security Audit Report

Socket

1. Contents

2. General Information

This report contains information about the results of the security audit of the Socket.Tech (hereafter referred to as "Customer") smart contracts, conducted by Decurity in the period from 04/15/2024 to 04/16/2024.

2.1. Introduction

Tasks solved during the work are:

- Review the protocol design and the usage of 3rd party dependencies,
- Audit the contracts implementation,
- Develop the recommendations and suggestions to improve the security of the contracts.

2.2. Scope of Work

The audit scope included the contracts in the following repository: https://github.com/SocketDotTech/socket-ll-contracts/pull/232. Initial review was done for the commit a418bce07f5781dd5478164396dbb2041c230235. The remediation review was done for the commit a67d7e86b62c456e0d7e0fd7b87a6d28659839ef.

The following contracts have been tested:

- HopCctpImplL2

2.3. Threat Model

The assessment presumes the actions of an intruder who might have the capabilities of any role (an external user, token owner, token service owner, or a contract).

The main possible threat actors are:

- User,
- Protocol owner,
- Relayer,

- Token owner/contract.

2.4. Weakness Scoring

An expert evaluation scores the findings in this report, and the impact of each vulnerability is calculated based on its ease of exploitation (based on the industry practice and our experience) and severity (for the considered threats).

2.5. Disclaimer

Due to the intrinsic nature of the software and vulnerabilities and the changing threat landscape, it cannot be generally guaranteed that a certain security property of a program holds.

Therefore, this report is provided “as is” and is not a guarantee that the analyzed system does not contain any other security weaknesses or vulnerabilities. Furthermore, this report is not an endorsement of the Customer’s project, nor is it an investment advice.

That being said, Decurity exercises the best effort to perform its contractual obligations and follow the industry methodologies to discover as many weaknesses as possible and maximize the audit coverage using limited resources.

3. Summary

As a result of this work, we haven't discovered any exploitable security issues.

The other suggestions included fixing the low-risk issues and some best practices (see Security Process Improvement).

3.1. Suggestions

The table below contains the discovered issues, their risk level, and their status as of 16 April, 2024.

Table. Discovered weaknesses

Issue	Contract	Risk Level	Status
Broken tests	test/solidity/bridges/hop/l2/HopCCTPL2-USDCE.t.sol	Info	Fixed
Unnecessary public variable	src/bridges/hop/l2/HopCctpImpl.sol	Info	Fixed
Lack of swapAndBridge function selector	src/bridges/hop/l2/HopCctpImpl.sol	Info	Fixed
Redundant imports	src/bridges/hop/l2/HopCctpImpl.sol	Info	Fixed

4. General Recommendations

This section contains general recommendations on how to improve the overall security level.

The Findings section contains technical recommendations for each discovered issue.

4.1. Security Process Improvement

The following is a brief long-term action plan to mitigate further weaknesses and bring the product security to a higher level:

- Keep the whitepaper and documentation updated to make it consistent with the implementation and the intended use cases of the system,
- Perform regular audits for all the new contracts and updates,
- Ensure the secure off-chain storage and processing of the credentials (e.g. the privileged private keys),
- Launch a public bug bounty campaign for the contracts.

5. Findings

5.1. Broken tests

Risk Level: Info

Status: Fixed in the commit ffcbf97d.

Contracts:

- test/solidity/bridges/hop/l2/HopCCTPL2-USDCE.t.sol

Description:

The tests are intended for the Polygon network but the `ETHEREUM_RPC` environment variable is used making the tests non-functional.

Remediation:

Use the correct RPC URL.

5.2. Unnecessary public variable

Risk Level: Info

Status: Fixed in the commit ae76df2d.

Contracts:

- src/bridges/hop/l2/HopCctpImpl.sol

Location: Lines: 26.

Description:

There is an unnecessary public getter for the `UINT_MAX` variable which causes additional bytecode creation and gas costs.

Remediation:

Replace the public constant with private immutable.

5.3. Lack of swapAndBridge function selector

Risk Level: Info

Status: Fixed in the commit a67d7e86.

Contracts:

- src/bridges/hop/l2/HopCctpImpl.sol

Description:

The contract `HopCctpImplL2` does not contain an immutable variable with the selector of the function `swapAndBridge`. However, other implementations do include it.

Remediation:

Consider adding an immutable variable with this selector.

5.4. Redundant imports

Risk Level: Info

Status: Fixed in the commit 2ff47a8e.

Contracts:

- src/bridges/hop/l2/HopCctpImpl.sol

Location: Lines: 9.

Description:

Library `console.sol` is imported in `HopCctpImpl` contract. This library is used for testing purposes. It is recommended to delete this dependency in production environment.

Remediation:

Consider deleting console imports.

6. Appendix

6.1. About us

The Decurity team consists of experienced hackers who have been doing application security assessments and penetration testing for over a decade.

During the recent years, we've gained expertise in the blockchain field and have conducted numerous audits for both centralized and decentralized projects: exchanges, protocols, and blockchain nodes.

Our efforts have helped to protect hundreds of millions of dollars and make web3 a safer place.