decurity

Smart Contract Security Audit Report

Socket Bungee CCTP Audit

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 16/04/2025 to 18/04/2025.

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/marketplace. Initial review was done for the commit f79abe25

The following contracts have been tested:

- src/routers/CCTPV2RouterSingleOutput.sol

2.3. Threat Model

The assessment presumes actions of an intruder who might have capabilities of any role (an external user, token owner, token service owner, a contract). The centralization risks have not been considered upon the request of the Customer.

The main possible threat actors are:

- User,
- Protocol owner,
- Liquidity Token owner/contract.

The table below contains sample attacks that malicious attackers might carry out.

Table. Theoretically possible attacks

Attack	Actor
Contract code or data hijacking *Deploying a malicious contract or submitting malicious data*	Contract owner Token owner
Financial fraud *A malicious manipulation of the business logic and balances, such as a re-entrancy attack or a flash loan attack*	Anyone
Attacks on implementation *Exploiting the weaknesses in the compiler or the runtime of the smart contracts*	Anyone

2.4. Weakness Scoring

An expert evaluation scores the findings in this report, an 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 best effort to perform their contractual obligations and follow the industry methodologies to discover as many weaknesses as possible and maximize the audit coverage using the limited resources.

3. Summary

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

The other suggestions included fixing the info-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 April 18, 2025.

Table. Discovered weaknesses

Issue	Contract	Risk Level	Status
Inconsistent error handling	src/routers/CCTPV2RouterSingleOutput.sol	Info	Not fixed
Lack of event	src/routers/CCTPV2RouterSingleOutput.sol	Info	Not fixed
Unused mapping	src/routers/CCTPV2RouterSingleOutput.sol	Info	Not fixed

4. General Recommendations

This section contains general recommendations on how to improve 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. Inconsistent error handling

Risk Level: **Info**

Status: Not fixed

Contracts:

- src/routers/CCTPV2RouterSingleOutput.sol

Description:

The CCTPV2Router contract mixes different error handling approaches, using both require statements and custom errors inconsistently throughout the code. This leads to inconsistent error messages and gas usage patterns.

```
// Line 149: Using custom error
if (receiverContract == address(0)) revert InvalidReceiver();

// Line 203: Using require statement
require(msgReceived.expiry > 0);

// Line 302: Using require statement again
require(msgReceived.expiry > 0);
```

Remediation:

Consider standardizing error handling by consistently using custom errors throughout the contract. This improves gas efficiency and provides clearer error messages for developers and users.

5.2. Lack of event

Risk Level: **Info**

Status: Not fixed

Contracts:

- src/routers/CCTPV2RouterSingleOutput.sol

Description:

The _fulfil() function does not emit an event like the _withdrawRequestOnDestination() function does. This may be necessary to track onchain activity.

Remediation:

Consider adding an event for the _fulfil() function calls.

5.3. Unused mapping

Risk Level: **Info**

Status: Not fixed

Contracts:

- src/routers/CCTPV2RouterSingleOutput.sol

Description:

The CCTPV2Router contract declares a mapping msgParams to track message parameters received from CCTP, but it is never used throughout the contract's implementation.

Remediation:

Consider removing the unused msgParams mapping from the contract.

6. Appendix

6.1. About us

The [Decurity](https://decurity.io) 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.