Smart Contract Security Audit Report

Socket (Mantle Integration)

1. Contents

1. Contents 2

2. General Information 3

2.1. Introduction 3
2.2. Scope of Work 3
2.3. Threat Model 3
2.4. Weakness Scoring 4
2.5. Disclaimer 4

3. Summary 5

3.1. Suggestions 5

4. General Recommendations 6

4.1. Security Process Improvement 6

5. Findings 7

5.1. No `MNT_TOKEN` check in `swapAndBridge` 7
5.2. Extra data parameter isn't passed in the depositMNTTo call 7
5.3. Incorrect selectors 8
5.4. Wrong variable names 8
5.5. Tests not working 9
5.6. Inconsistent event ordering 9

6. Appendix 11

6.1. About us 11

2. General Information

This report contains information about the results of the security audit of the Socket.Tech (hereafter referred to as "Customer") Mantle route smart contract, conducted by Decurity in the period from 11/25/2024 to 11/26/2024.

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 following smart contracts (commit e26e9e69613c8de6ff0d4c4faa9cb5518b155253):

- https://github.com/SocketDotTech/socket-ll-contracts/tree/feat/mantle-mnt-bridge

The retest was done for the commit 8e3470d1e0445d9da8efa93e9cbe27463cc0c810.

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 28 November, 2024.

Table. Discovered weaknesses

Issue	Contract	Risk Level	Status
No `MNT_TOKEN` check in `swapAndBridge`	src/bridges/mantle/MantleNative.sol	**Medium**	Fixed
Extra data parameter isn't passed in the depositMNTTo call	src/bridges/mantle/MantleNative.sol	**Info**	Fixed
Incorrect selectors	src/bridges/mantle/MantleNative.sol	**Info**	Fixed
Wrong variable names	src/bridges/mantle/MantleNative.sol	**Info**	Fixed
Tests not working	test/solidity/bridges/mantle/MantleEth.t.sol test/solidity/bridges/mantle/MantleUSDC.t.sol	**Info**	Fixed
Inconsistent event ordering	src/bridges/mantle/MantleNative.sol	**Info**	Acknowledged

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. No `MNT_TOKEN` check in `swapAndBridge`

Risk Level: Medium

Status: Fixed in the commit 7eadfaa2.

Contracts:

- src/bridges/mantle/MantleNative.sol

Description:

There's no check for `MNT_TOKEN` address in the `swapAndBridge` function which leads to `depositERC20To` being called, so `L1StandardBridge` will revert (see line 813 in https://vscode.blockscan.com/ethereum/0xb4133552ba49dfb60da6eb5ca0102d0f94ce071f).

Remediation:

Add the check similar to the other functions.

5.2. Extra data parameter isn't passed in the depositMNTTo call

Risk Level: Info

Status: Fixed in the commit 4301ff6a.

Contracts:

- src/bridges/mantle/MantleNative.sol

Location: Lines: `159, 291`.

Description:

When `depositMNTTo` is called, the `data` parameter is not used, "0x" is passed instead. The extra data is used in the event emitting and could be useful for the off-chain monitoring.

Remediation:

Pass the extra data, like when calling `depositERC20To`.

5.3. Incorrect selectors

Risk Level: Info

Status: Fixed in the commit ba81ce53.

Contracts:

- src/bridges/mantle/MantleNative.sol

Description:

The `bridgeERC20To`, `bridgeNativeTo`, and `swapAndBridge` selectors in the immutables are incorrect and don't match actual function signatures.

Remediation:

The correct selectors should be (note also how OPTIMISM should be replaced with MANTLE):

```
bytes4 public immutable NATIVE_MANTLE_ERC20_EXTERNAL_BRIDGE_FUNCTION_SELECTOR
= bytes4(
    keccak256(

"bridgeERC20To(address,address,uint32,bytes32,uint256,address,uint256,bytes)"
    )
);

bytes4 public immutable NATIVE_MANTLE_NATIVE_EXTERNAL_BRIDGE_FUNCTION_SELECTOR
= bytes4(
    keccak256(
        "bridgeNativeTo(address,uint32,uint256,uint256,bytes32,bytes)"
    )
);


bytes4 public immutable NATIVE_MANTLE_SWAP_BRIDGE_SELECTOR = bytes4(
    keccak256(

"swapAndBridge(uint32,bytes,(bytes32,address,uint256,uint32,address,bytes))"
    )
);
```

5.4. Wrong variable names

Risk Level: Info

Status: Fixed in the commit baee0f7a.

Contracts:

- src/bridges/mantle/MantleNative.sol

Description:

The variables and comments in the code mention Optimism instead of Mantle.

Remediation:

Rename the variables and rewrite the comments.

5.5. Tests not working

Risk Level: Info

Status: Fixed in the commit ba81ce53.

Contracts:

- test/solidity/bridges/mantle/MantleEth.t.sol
- test/solidity/bridges/mantle/MantleUSDC.t.sol

Description:

The tests are not functional.

Remediation:

Finish the test development and coverage.

5.6. Inconsistent event ordering

Risk Level: Info

Status: Comment by the Customer: Regarding the event ordering, we have kept that way to save gas by avoiding put events on each conditions.

Contracts:

- src/bridges/mantle/MantleNative.sol

Description:

The `SocketBridge` event is emitted before the bridging in `bridgeAfterSwap` and after the bridging in `bridgeERC20To`.

```
// bridgeERC20To
L1StandardBridge(customBridgeAddress).depositERC20To(...);
emit SocketBridge(...);  // Event after deposit

// vs bridgeAfterSwap
emit SocketBridge(...);  // Event before deposit
L1StandardBridge(customBridgeAddress).depositETHTo{value: amount}(...);
```

This can affect off-chain monitoring infrastructure.

Remediation:

Emit the event at the end of the function.

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.