CANTINA

MakerDAO: Op Token Bridge

Security Review

Cantina Managed review by:

Christoph Michel, Lead Security Researcher

M4rio.eth, Security Researcher

September 9, 2024

Contents

1 Introduction

1.1 About Cantina

Cantina is a security services marketplace that connects top security researchers and solutions with clients. Learn more at cantina.xyz

1.2 Disclaimer

Cantina Managed provides a detailed evaluation of the security posture of the code at a particular moment based on the information available at the time of the review. While Cantina Managed endeavors to identify and disclose all potential security issues, it cannot guarantee that every vulnerability will be detected or that the code will be entirely secure against all possible attacks. The assessment is conducted based on the specific commit and version of the code provided. Any subsequent modifications to the code may introduce new vulnerabilities that were absent during the initial review. Therefore, any changes made to the code require a new security review to ensure that the code remains secure. Please be advised that the Cantina Managed security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.3 Risk assessment

Severity	Description
Critical	*Must* fix as soon as possible (if already deployed).
High	Leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
Medium	Global losses <10% or losses to only a subset of users, but still unacceptable.
Low	Losses will be annoying but bearable. Applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.
Gas Optimization	Suggestions around gas saving practices.
Informational	Suggestions around best practices or readability.

1.3.1 Severity Classification

The severity of security issues found during the security review is categorized based on the above table. Critical findings have a high likelihood of being exploited and must be addressed immediately. High findings are almost certain to occur, easy to perform, or not easy but highly incentivized thus must be fixed as soon as possible.

Medium findings are conditionally possible or incentivized but are still relatively likely to occur and should be addressed. Low findings a rare combination of circumstances to exploit, or offer little to no incentive to exploit but are recommended to be addressed.

Lastly, some findings might represent objective improvements that should be addressed but do not impact the project's overall security (Gas and Informational findings).

2 Security Review Summary

The Maker Protocol, also known as the Multi-Collateral Dai (MCD) system, allows users to generate Dai (a decentralized, unbiased, collateral-backed cryptocurrency soft-pegged to the US Dollar) by leveraging collateral assets approved by the Maker Governance, which is the community organized and operated process of managing the various aspects of the Maker Protocol.

From Aug 27th to Aug 28th the Cantina team conducted a review of op-token-bridge on commit hash 4b6cd2a3.

The Cantina team reviewed MakerDAO's op-token-bridge changes holistically on commit hash bae891c1 and determined that all issues were resolved and no new issues were identified.

The team identified a total of **3** issues in the following risk categories:

- Critical Risk: 0
- High Risk: 0
- Medium Risk: 0
- Low Risk: 0
- Gas Optimizations: 0
- Informational: 3

3 Findings

3.1 Informational

3.1.1 Require error typo

Severity: Informational

Context: L2TokenBridgeSpell.sol#L70

Description: The following require error is wrong as we do not have a Gateway component.

```
require(address(l2Bridge) == l2Bridge_, "L2TokenBridgeSpell/l2-gateway-mismatch");
```

Recommendation: Consider changing it to `L2TokenBridgeSpell/l2-bridge-mismatch`

Maker: Fixed in commit bae891c1.

Cantina Managed: Fixed.

3.1.2 Wrong `l1ToL2Token` verification on `L2TokenBridge`

Severity: Informational

Context: L2TokenBridge.sol#L126

Description: The `L2TokenBridge` checks that the `localToken` and `remoteToken` define a valid mapping by checking:

```
require(_remoteToken != address(0) && l1ToL2Token[_remoteToken] == _localToken, "L2TokenBridge/invalid-token");
```

However, as the default mapping value of `l1ToL2Token[address] = 0`, choosing any `_localToken == 0` and `_remoteToken > 0` for an unset token mapping will pass this check. The error is caught once the burn token call is attempted on `address(0)`, correctly reverting the transaction and reducing the impact of the wrong check.

Recommendation: To avoid user errors, it should check that the mapping exists, i.e., checking the mapping value, not the key, to be non-zero:

```
- require(_remoteToken != address(0) && l1ToL2Token[_remoteToken] == _localToken,
↪  "L2TokenBridge/invalid-token");
+ require(_localToken != address(0) && l1ToL2Token[_remoteToken] == _localToken, "L2TokenBridge/invalid-token" ↲
);
```

***Note:** that this is done correctly in L1TokenBridge.*

Maker: Fixed in commit bae891c1.

Cantina Managed: Fixed.

3.1.3 Wrong documentation for `L2TokenBridge.finalizeBridgeERC20`

Severity: Informational

Context: L2TokenBridge.sol#L188

Description: The documentation for `L2TokenBridge.finalizeBridgeERC20` reads:

> Finalizes an ERC20 bridge on L2. Can only be triggered by the L2TokenBridge.

Recommendation: It should read: "Can only be triggered by the L1TokenBridge."

Maker: Fixed in commit bae891c1.

Cantina: Verified.