CANTINA

MakerDAO: Endgame Updates

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.

On Aug 23rd the Cantina team reviewed MakerDAO's lockstate changes holistically on commit hash 8b16eac9

The team identified **1** issue in the following risk category:

- Critical Risk: 0
- High Risk: 0
- Medium Risk: 0
- Low Risk: 1
- Gas Optimizations: 0
- Informational: 0

The aforementioned issues were resolved and no new issues were identified with respect to the previous audited commit hash de66d6fc.

3 Findings

3.1 Low Risk

3.1.1 `lock`ing and `free`ing from `LockstakeEngine` can fail if `ilk.dust` increases

Severity: Low Risk

Context: Global Scope

Description: The `lock` and `free` functions add/remove collateral from the urn using `vat.frob(ilk, urn, urn, address(0), ±int256(wad), 0);`. This function will fail if the `urn.art > 0` && `urn.art * rate < ilk.dust`. It's possible that `ilk.dust` increased and `urn.art` does not each this new threshold, resulting in `vat.frob` calls reverting.

Users cannot lock more collateral to their urn without first changing their debt, neither can they withdraw collateral even if the resulting position would still be healthy.

Recommendation: Consider using `vat.grab(ilk, urn, urn, address(0), int256(wad), 0);` and checking `vat.live` for `_lock`, and implementing a similar function to `vat.frob` for `dart=0` but without the `"Vat/dust"` check.

Maker: We believe this should keep consistency with the rest of the ilks and avoid a special behavior. In other ilks the user can't lock in this situation and the debt check is applied (even if that's not necessary). The locking on the regular `lock` is a user action, while manipulating the user vault in `onRemove` is a system action, and is new for Lockstake.

Cantina Managed: Acknowledged.