PUBLIC

Code Assessment

of the Rate Limited Flapper Smart Contracts

February 18, 2022

Produced for

MAKER

by

CHAINSECURITY

Contents

1 Executive Summary

Dear all

Thank you for trusting us to help MakerDAO with this security audit. Our executive summary provides an overview of subjects covered in our audit of the latest reviewed contracts of Rate Limited Flapper according to Scope to support you in forming an opinion on their security risks.

MakerDAO added new functionality to the existing flapper contract: The amount of DAI under auction at the same time can now be limited.

The most critical subjects covered in our audit are functional correctness of the changed code and the impact of the change on the existing system.

In summary, we find that the introduced change works correctly and does not introduce a security risk. No issue was uncovered during the review.

It is important to note that security audits are time-boxed and cannot uncover all vulnerabilities. They complement but don't replace other vital measures to secure a project.

The following sections will give an overview of the system, our methodology, the issues uncovered and how they have been addressed. We are happy to receive questions and feedback to improve our service.

Sincerely yours,

ChainSecurity

1.1 Overview of the Findings

Below we provide a brief numerical overview of the findings and how they have been addressed.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	0

2 Assessment Overview

In this section, we briefly describe the overall structure and scope of the engagement, including the code commit which is referenced throughout this report.

2.1 Scope

The assessment was performed on the source code files inside the Rate Limited Flapper repository based on the documentation files. The table below indicates the code versions relevant to this report and when they were received.

V	Date	Commit Hash	Note
1	15 February 2022	5a67f7e49cfe4247cc9b8314ddb31943d6131c0e	Initial Version

For the solidity smart contracts, the compiler version `0.5.12` was chosen since the previous version of flapper used that compiler. The diff of file flap.sol is in scope (compared to `95e7169cf85c7dcdf83cae56456dc0dfd8f2c3bf`).

2.1.1 *Excluded from scope*

All files besides flap.sol.

2.2 System Overview

This system overview describes the initially received version (**Version 1**) of the contracts as defined in the Assessment Overview.

MakerDAO implemented a modification to the current flapper contract that allows limiting the amount of DAI that can be auctioned off at the same time.

2.2.1 *Flapper overview*

The flapper contract is used for auctions of surplus DAI that was generated from the stability fee in the Maker system. The bidders bid with MKR tokens that will be burned after an auction has ended. The contract offers following functionality:

- The governance function `file()` setting global auction values such as `beg` (minimum bid increase), `ttl` (the maximum between two bids on an auction), and `tau` (the maximum auction duration).
- The authenticated function `cage` that cages the flapper (used to activate ESM for this contract).
- The default `rely()` / `deny` authentication in the Maker system.
- `kick`: Start an auction.
- `tick`: Restart an auction if no bids were made.
- `tend`: Bid on an auction.
- `deal`: Finish the auction.
- `yank`: In case the ESM is activated, end the `tend` phase auctions.

2.2.2 Changes in the flapper

Before, the amount of DAI that could be in auctions at the same time was not restricted. The introduced changes allow to set a limit on the amount of DAI allowed in the Flapper. Hence, two new state variables have been introduced

- `lid`: the maximum amount of DAI in auctions in `rad`
- and `fill`: the current amount of DAI in auctions in `rad`

for which `fill <= lid` should hold when a new auction has started. Note that the invariant could be temporarily violated when `lid` is set below `fill`. Thus, all functions moving DAI in and out of the Flapper (`kick()` and `deal()`) were modified to enforce that invariant and to update `fill`.

Additionally, `file()` is extended to allow changing `lid`.

3 Limitations and use of report

Security assessments cannot uncover all existing vulnerabilities; even an assessment in which no vulnerabilities are found is not a guarantee of a secure system. However, code assessments enable the discovery of vulnerabilities that were overlooked during development and areas where additional security measures are necessary. In most cases, applications are either fully protected against a certain type of attack, or they are completely unprotected against it. Some of the issues may affect the entire application, while some lack protection only in certain areas. This is why we carry out a source code assessment aimed at determining all locations that need to be fixed. Within the customer-determined time frame, ChainSecurity has performed an assessment in order to discover as many vulnerabilities as possible.

The focus of our assessment was limited to the code parts defined in the engagement letter. We assessed whether the project follows the provided specifications. These assessments are based on the provided threat model and trust assumptions. We draw attention to the fact that due to inherent limitations in any software development process and software product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which itself cannot be free from any error or failures. These preconditions can have an impact on the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure which adds further inherent risks as we rely on the correct execution of the included third-party technology stack itself. Report readers should also take into account that over the life cycle of any software, changes to the product itself or to the environment in which it is operated can have an impact leading to operational behaviors other than those initially determined in the business specification.

4 Terminology

For the purpose of this assessment, we adopt the following terminology. To classify the severity of our findings, we determine the likelihood and impact (according to the CVSS risk rating methodology).

- *Likelihood* represents the likelihood of a finding to be triggered or exploited in practice
- *Impact* specifies the technical and business-related consequences of a finding
- *Severity* is derived based on the likelihood and the impact

We categorize the findings into four distinct categories, depending on their severity. These severities are derived from the likelihood and the impact using the following table, following a standard risk assessment procedure.

Likelihood	Impact		
	High	Medium	Low
High	Critical	High	Medium
Medium	High	Medium	Low
Low	Medium	Low	Low

As seen in the table above, findings that have both a high likelihood and a high impact are classified as critical. Intuitively, such findings are likely to be triggered and cause significant disruption. Overall, the severity correlates with the associated risk. However, every finding's risk should always be closely checked, regardless of severity.

5 Findings

In this section, we describe our findings. The findings are split into these different categories:

Below we provide a numerical overview of the identified findings, split up by their severity.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	0