Sky: Lockstake

Security Review

Cantina Managed review by:
Christoph Michel, Lead Security Researcher
Mario.eth, Lead Security Researcher

August 18, 2025

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

Sky Protocol is a decentralised protocol developed around the USDS stablecoin.

From Jul 28th to Aug 8th the Cantina team conducted a review of lockstake on commit hash 37c18fe3. The team identified a total of **3** issues in the following risk categories:

Issues Found

Severity	**Count**	**Fixed**	**Acknowledged**
Critical Risk	0	0	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	1	1	0
Gas Optimizations	0	0	0
Informational	2	1	1
Total	**3**	**2**	**1**

The Cantina Managed team reviewed Sky's `lockstake` holistically on commit hash d4dbe6ea and concluded that all the issues were addressed and no new vulnerabilities were identified.

3 Findings

3.1 Low Risk

3.1.1 `LockstakeInit.updateClipper` is missing `stUSDS.rely(clipper)`

Severity: Low Risk

Context: LockstakeInit.sol#L311

Description: The `stUsds` contract needs to `rely` the new `clipper` so it can call `stUsds.cut(..)` upon auctions that end with bad protocol debt.

This call is missing in `updateClipper`, even though the function seems to support migrating from an activated `clipper`:

```
// stopped could be 0, meaning new clipper should also be activated!
se.clipper.file("stopped", se.oldClipper.stopped());
// copy over current mom state
if (se.oldClipper.wards(address(clipperMom)) == 1) {
    se.clipper.rely(address(clipperMom));
}
```

If the init spell ends up activating the new clipper with `.file("stopped", 0)`, it won't be fully operational. Bad debt liquidations will revert in `newClipper.take -> stUsds.cut`.

Recommendation: Consider performing the `stUSDS.rely(clipper)` call in `updateClipper`. In addition, the team mentioned that the (old) clipper will be deactivated (`stopped = 3`) during the migration, same for the new clipper for the initial period of the `stUSDS` launch. As an alternative, change `updateClipper` to always deactivate the new clipper and consider adding another function that activates it to the deploy scripts.

Sky: Fixed in commit 43662905.

Cantina Managed: Fix verified. Script now checks the old clipper is deactivated and has no active auctions before running. `updateClipper` and `removeClipper` have been merged, new `enableLiquidations` function has been added.

3.2 Informational

3.2.1 Mixed usage of `DAI` and `USDS` in comments

Severity: Informational

Context: LockstakeClipper.sol

Description: Many comments within the `LockstakeClipper` retained from the Maker code still talk about "DAI" - e.g., *"get DAI from caller," "remaining DAI target", "Don't collect more than tab of DAI"* but we also have references to USDS `uint256 tab;  // Usds to raise rad]`.

Using a mixed reference can be confusing for the readers.

Recommendation: Consider sticking to one naming convention.

Sky: Acknowledged, decided to keep as it is to avoid further changes and a bigger diff at this stage.

Cantina Managed: Acknowledged by the client.

3.2.2 Misleading comment in `LockstakeClipper.kick` about triggering bad debt accounting

Severity: Informational

Context: LockstakeClipper.sol#L279

Description: Usually, **bad debt** refers to losses to the protocol, i.e., what is used for the `cut` call in `clipper.take`. In `clipper.kick` (which starts the liquidation), one does not know yet if there'll be bad debt but the comment talks about triggering *bad debt accounting*.

```
// Trigger bad debt accounting (will update line accordingly)
if (cuttee != address(0)) { CutteeLike(cuttee).drip(); }
```

This comment is therefore slightly misleading or could be misinterpreted.

Recommendation: Consider rephrasing it, for example:

```
// Trigger new auction accounting (will update line accordingly)
if (cuttee != address(0)) { CutteeLike(cuttee).drip(); }
```

Sky: This was done in commit c217ac96. And the final version in commit 43662905.

Cantina Managed: Fix verified.