Sky: Star Guard
Security Review

Cantina Managed review by:

Mario.eth, Lead Security Researcher
Xmxanuel, Lead Security Researcher

October 23, 2025

Contents

1 Introduction **2**
- 1.1 About Cantina . . . 2
- 1.2 Disclaimer . . . 2
- 1.3 Risk assessment . . . 2
  - 1.3.1 Severity Classification . . . 2

2 Security Review Summary **3**

3 Findings **4**
- 3.1 Informational . . . 4
  - 3.1.1 Payable `SubProxy.exec` is not supported by `StarGuard` . . . 4
  - 3.1.2 `plot` can replace an already queued spell . . . 4
  - 3.1.3 The current design does not offer multiple spells for the same SubProxy . . . 4
  - 3.1.4 Incorrect `README.md` flow diagram for star spell whitelisting . . . 5
  - 3.1.5 `StarGuard` can't `deny` itself on the `subProxy` in case of a `StarGuard` migration event . . . 5
  - 3.1.6 `StarGuard.exec` doesn't returned `bytes memory out` from `subProxy.exec` . . . 6
  - 3.1.7 `maxDelay` can be set to zero allowing execution only in the same block as the `plot` call 6

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 level	Impact: High	Impact: Medium	Impact: Low
Likelihood: high	Critical	High	Medium
Likelihood: medium	High	Medium	Low
Likelihood: low	Medium	Low	Low

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 are 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 Oct 6th to Oct 17th the Cantina team conducted a review of star-guard on commit hash 7398ffb2. The review focused on the following scope:

- README.md.
- StarGuard.sol.
- StarGuardInit.sol.

The team identified a total of **7** issues:

Issues Found

Severity	Count	Fixed	Acknowledged
Critical Risk	0	0	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	0	0	0
Gas Optimizations	0	0	0
Informational	7	2	5
Total	**7**	**2**	**5**

The Cantina Managed team reviewed Sky's `star-guard` holistically on commit hash 52239d71 (tag v1.0.1) and concluded that all findings were addressed and no new vulnerabilities were identified.

3 Findings

3.1 Informational

3.1.1 Payable `SubProxy.exec` is not supported by `StarGuard`

Severity: Informational

Context: StarGuard.sol#L219

Description: The `exec` function in the `SubProxy` is payable and can receive ETH. This feature is not supported by the `StarGuard.exec` function.

Recommendation: If a payable support is needed for `StarGuard`, the cleanest way to add support would be to extend the `SpellData` by a `value` variable.

The `value` would be set by `plot`, along with the following changes in `exec`

```
require(address(this).balance >= spellDataCopy.value, "StarGuard/insufficient-balance");
delete spellData;
subProxy.exec{value: spellDataCopy.value}(
    spellDataCopy.addr,
    abi.encodePacked(StarSpellLike.execute.selector)
);
```

In addition, a `receive` function would be required. This would ensure the permissionless caller of the `exec` method can't modify the used `value` for the `subProxy` call.

The `MCD_PAUSE_PROXY` remains a `ward` on the `subProxy` together with the `StarGuard`, so it would be still possible to transfer ETH to the `subProxy`.

Sky: We don't want to overcomplicate the StarGuard design by implementing payable methods, handling non-standard interfaces, custom calldata, etc..., for which we currently have no practical use cases.

Cantina Managed: Acknowledged.

3.1.2 `plot` can replace an already queued spell

Severity: Informational

Context: StarGuard.sol#L181-L186

Description: The `plot` function overwrites any existing scheduled spell without checking if one is already pending. Calling `plot` while a spell is still queued silently replaces `spellData` (address, tag, deadline).

Recommendation: Add an explicit check to prevent accidental overwrite by reverting if `spellData.addr != address(0)` so operators must call `drop()` before plotting a new spell;

An alternative would be to keep plot as-is but enforce the "no pending spell" check in the Core Spell before calling `plot`, and document this requirement.

A second alternative would be to assume an auto-drop in this case and emit `Drop` before replacing the `spellData`

Sky: Fixed in commit a7bf3d1.

Note: since the original *overwrite* logic was intentional, we went with the second alternative:

> To assume an auto-drop in this case and emit `Drop` before replacing the `spellData`

Cantina Managed: Fixed.

3.1.3 The current design does not offer multiple spells for the same SubProxy

Severity: Informational

Context: StarGuard.sol#L73, StarGuard.sol#L181-L186

Description: With one StarGuard instance per SubProxy and a single `spellData` slot, only one spell can be scheduled at a time for that `SubProxy`. This design prevents staging multiple independent spells concurrently, even when they are unrelated or can safely execute in any order.

Recommendation: If this would be a desired flow in the future, an option to implement would be to extend `StarGuard` storage to a bounded mapping keyed by (addr, tag) with explicit plot/drop/exec(id) with ordering/priority enforced in the `SpellData`.

Sky: Currently this is by design, one single Spell can be scheduled per `SubProxy`.

Cantina Managed: Acknowledged.

3.1.4 Incorrect `README.md` flow diagram for star spell whitelisting

Severity: Informational

Context: README.md#L19

Description: The `README.md` diagram describes the new flow when the new `StarGuard` pattern will be used to execute star spells.

Flow Diagram from the README.md

```
1st transaction: Anyone    Core Spell    StarGuard A  whitelists   SubProxy A    Star Spell A1
                         StarGuard B  whitelists   SubProxy B    Star Spell B1
2nd transaction: Anyone            StarGuard A   executes    SubProxy A    Star Spell A1
3rd transaction: Anyone            StarGuard B   executes    SubProxy B    Star Spell B1
```

This flow is incorrect.

For the whitelisting with `StarGuard.plot`, the function `plot` function won't call any whitelisting on the `SubProxy`.

Recommendation: The correct digram is:

```
  1st transaction: Anyone    Core Spell    StarGuard A.plot(Star Spell A1)
                                             StarGuard B.plot(Star Spell B1)
  2nd transaction: Anyone            StarGuard A.exec()    SubProxy A    Star Spell A1
  3rd transaction: Anyone            StarGuard B.exec()    SubProxy B    Star Spell B1
```

Sky: Fixed in commit b0605aa

Cantina Managed: Fixed.

3.1.5 `StarGuard` can't `deny` itself on the `subProxy` in case of a `StarGuard` migration event

Severity: Informational

Context: StarGuard.sol#L221

Description: After the `subProxy.exec` call in `exec`, the `StarGuard` checks whether it is still a ward on the `subProxy`:

```
require(subProxy.wards(address(this)) == 1, "StarGuard/subProxy-owner-change");
```

This check is required because the `starSpell` is executed via `delegate_call` from the `subProxy` and could modify the ward storage.

However, this safety check also prevents legitimate self-removals of a ward. For example, during a migration to a new StarGuard version.

In such a scenario, if the `StarGuard` couldn't remove itself. The new `StarGuard` version would have to remove the old one.

In the current setup, the `MCD_PAUSE_PROXY` remains a ward on all `subProxies` and could execute the migration.

Recommendation: In case the `MCD_PAUSE_PROXY` control over the `subProxies` is ever removed, consider adding methods to forward ward control from the `StarGuard`.

```
function relySubProxy(address usr) external auth {
    subProxy.rely(usr);
}

function denySubProxy(address usr) external auth {
    subProxy.deny(usr);
}
```

This would allow the a `StarGuard` function to remove itself as a `SubProxy` ward.

Sky: This module isn't meant to be the sole SubProxy owner. It only executes standard Star spells. Non-standard or same-block spells can still be executed directly via SubProxy as today.

Cantina Managed: Acknowledged.

3.1.6 `StarGuard.exec` doesn't returned `bytes memory out` from `subProxy.exec`

Severity: Informational

Context: StarGuard.sol#L219

Description: The `StarGuard.exec` function returns the `address` of the executed starSpell but not the return ed `bytes memory out` from the exec call in the `subProxy`.

The `subProxy.exec` returns a `bytes memory out`. See SubProxy.sol#L74.

Recommendation: Consider returning the `bytes memory out` and the `starSpell` address in the `StarGuard.exec`.

Sky: We don't think it is necessary: the standard spell interface we're requiring in the readme does not expect to return anything.

Cantina Managed: Acknowledged.

3.1.7 `maxDelay` can be set to zero allowing execution only in the same block as the `plot` call

Severity: Informational

Context: StarGuard.sol#L167

Description: The `maxDelay` defines the time window for execution a spell. It is possible to set the `maxDelay` to zero with a `file` call. If set to zero a star spell needs to be executed in the same block as the `plot` call.

It wouldn't be possible to successfully call `StarGuard.exec` later because the starSpell would immediately expire after the block.

Recommendation: If it should be not possible to set the `maxDelay` to zero add a require statement in the `file` method..

Sky: We intentionally allow `maxDelay = 0` since the spell can still be executed within the same block as the `plot` call by the `coreSpell`. We don't see any harm in this and already have a sanity check in the init library to cover it.

Cantina Managed: Acknowledged.