Security Review

Cantina Managed

SKY: INSTANT USDC ↔ USDS CONVERTER

Cantina Managed review by:
Christoph Michel, Lead Security Researcher
Xmxanuel, Lead Security Researcher

CANTINA

October 24, 2026

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 No-op `removeSwapper` emits `SwapperRemoved` . . . 4
  - 3.1.2 Generic `verify/` prefixes make verifier failures harder to trace . . . 4
  - 3.1.3 `VerifyInstantUsdcUsdsConverter` cannot distinguish the converter from a malicious contract implementing the same interface . . . 4
  - 3.1.4 Minor issues, Typos & Documentation . . . 5

1 Introduction

1.1 About Cantina

Cantina is a security platform that pairs a world-class network of security researchers with agentic AI to help teams building onchain find and fix real risk. Through managed reviews, competitions, and bounties, Cantina secures protocols before vulnerabilities can be exploited.
Learn more at cantina.security

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

On Sep 27th the Cantina team conducted a review of grove-periphery on commit hash 45121058. The team identified a total of **4** 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	4	0	4
Total	**4**	**0**	**4**

3 Findings

3.1 Informational

3.1.1 No-op `removeSwapper` emits `SwapperRemoved`

Severity: Informational.

Context: src/InstantUsdcUsdsConverter.sol#L180-L185.

Description: `removeSwapper` discards `_revokeRole`'s return value and emits `SwapperRemoved` even when no role was removed. For a target without `SWAPPER_ROLE`, OpenZeppelin's `_revokeRole` returns `false` and leaves storage unchanged. Event consumers that use this event as the revocation signal can therefore record a state change that did not occur. The committed `test_removeSwapper_onNonSwapperEmitsAndNoops` test demonstrates the path: `hasRole(SWAPPER_ROLE, target)` stays `false` while the custom event is emitted.

Recommendation: Use the boolean returned by `_revokeRole` before emitting, or revert when the target does not hold `SWAPPER_ROLE`:

```
if (_revokeRole(SWAPPER_ROLE, swapper)) {
    emit SwapperRemoved(swapper);
}
```

Sky: Acknowledged.

Cantina Managed: Acknowledged.

3.1.2 Generic `verify/` prefixes make verifier failures harder to trace

Severity: Informational.

Context: script/VerifyInstantUsdcUsdsConverter.s.sol#L17-L46.

Description: The shared verifier is declared as `VerifyInstantUsdcUsdsConverterBase`, but each failure string uses the shorter `verify/` prefix. The resulting revert reasons are less specific than the contract name and are harder to trace when the mainnet or custom verifier fails. We can see that the other contract `DeployInstantUsdcUsdsConverterMainnet` uses its name as the prefix in revert messages.

Recommendation: Use a consistent prefix such as `VerifyInstantUsdcUsdsConverterBase/lite-psm`, `VerifyInstantUsdcUsdsConverterBase/admin-role`.

Sky: Acknowledged.

Cantina Managed: Acknowledged.

3.1.3 `VerifyInstantUsdcUsdsConverter` cannot distinguish the converter from a malicious contract implementing the same interface

Severity: Informational.

Context: script/VerifyInstantUsdcUsdsConverter.s.sol#L32-L44.

Description: `_verify` only compares getter return values and never inspects the code at `CONVERTER_ADDRESS`. Any contract that returns the expected constants and `true` from `hasRole` passes and logs "verified", whatever its `swap` does.

Since the deployer is untrusted, this script is the check between a deployed address and the governance spell granting it a USDC allowance. an impostor that passes it can move the full allowance in a single `swap` call.

Recommendation: An additional check can confirm the correct bytecode as well by deploying locally in the verify script. Immutables are embedded in the runtime bytecode, so a reference instance built with the same constructor arguments has an identical codehash:

```
address ref = address(
    InstantUsdcUsdsConverterDeploy.deploy(admin, swapper, freezer, holder, litePsm,
    ↪  daiUsds)
);
require(converter_.codehash == ref.codehash, "verify/codehash");
```

Sky: Acknowledged.

Cantina Managed: Acknowledged.

3.1.4 Minor issues, Typos & Documentation

Severity: Informational.

Context: *(See each case below)*.

Description:

1. InstantUsdcUsdsConverter.sol#L91: The natspec for `swapper_` says "allowed to call the swaps", but only one swap function exists. Use `{swap}`, matching the `{removeSwapper}` reference on the next line.
2. InstantUsdcUsdsConverter.sol#L154-L162: If the PSM fee is active (`tin != 0`) and the converter is not whitelisted via `bud`, the swap reverts with `NotOneToOne`, the same error a misbehaving PSM produces, after the USDC has already been pulled and sold. Consider a dedicated error checked before any funds move:

```
if (!noFeeRoute && litePsm.tin() != 0) revert FeeActiveNotWhitelisted();
```

The `NotOneToOne` check should stay, as it verifies what the PSM actually paid. The same line also covers a halted PSM, where `tin` is `type(uint256).max`.

Recommendation: Consider addressing the mentioned issues.

Sky: Acknowledged.

Cantina Managed: Acknowledged.