CANTINA

Ondo Finance

Security Review

Cantina Managed review by:
R0bert, Lead Security Researcher

November 21, 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 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

Ondo's mission is to make institutional-grade financial products and services available to everyone.

From Nov 17th to Nov 18th the Cantina team conducted a review of rwa-internal on commit hash c79a762e. The team identified a total of **5** issues:

Issues Found

Severity	Count	Fixed	Acknowledged
Critical Risk	0	0	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	1	0	1
Gas Optimizations	0	0	0
Informational	4	2	2
Total	**5**	**2**	**3**

2.1 Scope

The security review had the following components in scope for rwa-internal on commit hash c79a762e:

```
├── contracts
│   └── rwaOracles
│       ├── ISyntheticSharesOracle.sol
│       └── SyntheticSharesOracle.sol
└── forge-tests
    └── rwaOracles
        └── SyntheticSharesOracle.t.sol
```

3 Findings

3.1 Low Risk

3.1.1 Contract deviations from Notion spec

Severity: Low Risk

Context: SyntheticSharesOracle.sol#L40

Description: Several behaviors differ from the stated design for the Chainlink-integrated sValue oracle:

- Pause lead time not enforced: `scheduleCorporateActionPause` allows immediate or any future pause start; there is no 24h-in-advance requirement.
- Freeze-at-pause-start support incomplete: `getSValue` only returns `(sValue, paused)` and does not include `pauseStartTime`, `pendingSValue`, or any price-at-pause-start. `assetData` is public, so callers can read `pauseStartTime/pendingSValue` directly, but there is no built-in ”freeze price at pause start” mechanism.
- Proportional drift over elapsed time not implemented: the code uses a flat per-update cap `allowedDriftBps` plus `driftCooldown`, not the ”min(1%, 1% * timeDelta/24h)” proportional rule from the spec.
- Non-reverting read not honored: `getSValue` reverts on unknown assets instead of returning a sentinel, contradicting the ”oracle should never revert execution” assumption (though acceptable if all assets are preconfigured and calls are gated).
- Pause duration floor too low for intended freeze: minimum pause can be set to 600s; spec expectation implies a stronger freeze window. Admin can reduce the safety window via `setMinimumPauseDuration`.
- Pause metadata not surfaced in `getSValue`: pending values and pause start are not in the primary read API. Consumers must read `assetData` directly. If the intended API is only `getSValue`, this omits needed context for off-chain logic.

Recommendation: Consider aligning the contract with the published spec: enforce a minimum lead time for pauses; expose `pauseStartTime` and `pendingSValue` (or price-at-pause-start) in the read API to support freeze semantics, implement proportional drift relative to elapsed time, make reads non-reverting with sentinel values for unknown assets (if required by ops), raise the minimum pause floor to match the desired freeze window and surface pause metadata in the primary read path or document reliance on `assetData`. Document any intentional deviations if kept.

Ondo Finance: Acknowledged. The original design document is somewhat out of date but we confirm that the current implementation reflects the intended and up to date spec.

Cantina Managed: Acknowledged.

3.2 Informational

3.2.1 Unused custom error

Severity: Informational

Context: ISyntheticSharesOracle.sol#L157-L158

Description: The interface declares an `UnknownAsset()` error but the `SyntheticSharesOracle` contract implementation never uses it.

Recommendation: Either remove the unused `UnknownAsset` error from the interface or use it consistently for missing-asset checks to align the ABI with the on-chain behavior.

Ondo Finance: Fixed in PR 500

Cantina Managed: Fix verified.

3.2.2 GetSValueBatch reverts on unknown asset

Severity: Informational

Context: SyntheticSharesOracle.sol#L116

Description: `getSValueBatch` calls `_getSValue` for each asset and will revert if any asset is missing (`AssetNotFound`). This makes the batch all-or-nothing. A single unknown asset aborts the entire call. If the intended behavior is to retrieve what is available and skip unknown entries, this revert pattern prevents partial results.

Recommendation: If partial results are desired, return a sentinel/flag per index (e.g., `sValue=0`, `paused=false` and a boolean `found`) instead of reverting, or pre-check existence and skip/zero-fill unknown assets while still completing the batch.

Ondo Finance: Acknowledged. Reverting `getSValueBatch` upon nonexistent asset is the desirable behavior. Updated the design doc to reflect this.

Cantina Managed: Acknowledged.

3.2.3 Pause flag is advisory only

Severity: Informational

Context: SyntheticSharesOracle.sol#L119

Description: Corporate-action pauses only toggle a boolean returned by `getSValue` and have no onchain enforcement on price usage. The read path surfaces `paused` but does not block consumption of `sValue`:

```
function _getSValue(address asset) internal view returns (uint128 sValue, bool paused) {
    Asset storage a = assetData[asset];
    if (a.sValue == 0) revert AssetNotFound();
    sValue = a.sValue;
    paused = _isPauseActive(a);
}
```

If integrators fetch only `sValue` and ignore `paused`, they will continue using stale or unverified multipliers during corporate events, defeating the pause intent. Because of this downstream systems that omit the pause check can ingest and act on prices that should be withheld during corporate actions.

Recommendation: Require downstream consumers to gate price usage on the `paused` flag and alert/fail closed when true. Alternatively, add onchain checks in calling contracts to revert or refuse price use when `paused` is true so accidental consumption is prevented.

Ondo Finance: Acknowledged. This is core to the design and will require downstream consumers to understand the meaning of ”paused” in this context.

Cantina Managed: Acknowledged.

3.2.4 Setter path cannot be frozen via params

Severity: Informational

Context: SyntheticSharesOracle.sol#L383

Description: The setter path always permits some upward drift because `allowedDriftBps` must be > 0; `setDriftParameters` rejects `allowedDriftBps == 0`. Operationally, this means you cannot freeze nominal updates via parameters without invoking the admin-driven pause/corporate-action flow. If the setter role is suspected compromised or you want to halt routine bumps, there is no parameter-only way to stop it.

Recommendation: Consider allowing `allowedDriftBps = 0` to freeze the nominal update path when desired, or add an explicit freeze flag for the setter role. Retain existing bounds checks otherwise.

Ondo Finance: Fixed in PR 501

Cantina Managed: Fix verified.