Sky: TimelockController v5.5.0 Basin scripts

Security Review

Cantina Managed review by:
Christoph Michel, Lead Security Researcher
M4rio.eth, Lead Security Researcher

June 18, 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**
- 2.1 Scope . . . 3

3 `TimelockController` v5.4.0 → v5.5.0 changes **4**

4 Basin Scripts **4**
- 4.1 Low Risk . . . 4
  - 4.1.1 `SetupJTRSYMorphoUsdtBasin.s.sol` refactor . . . 4
- 4.2 Informational . . . 4
  - 4.2.1 `REDEEMER_ROLE` is held by an issuer-controlled address, not by Basin governance . . . 4
  - 4.2.2 Third-party post-deployment steps required . . . 5
  - 4.2.3 Lingering deployer roles . . . 5
  - 4.2.4 Minor issues . . . 5
  - 4.2.5 Minor issues, Typos & Documentation . . . 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.

On May 28th the Cantina team conducted a review of grove-basin on commit hash ac77e496. The team identified a total of **6** 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	1	0
Gas Optimizations	0	0	0
Informational	5	2	3
Total	**6**	**3**	**3**

The Cantina Managed team reviewed Sky's grove-basin holistically on commit hash be55a5dd (tag v1.0.0) and concluded that all findings were addressed and no new vulnerabilities were identified.

2.1 Scope

The security review had the following components in scope for grove-basin on commit hash ac77e496:

TimelockController v5.4.0 $\rightarrow$ v5.5.0 differential review and the Basin script directory.

The correctness of hardcoded script addresses - including rate providers, issuer redeemers & redemption addresses, timelocks, and multisigs - was taken on faith from the project team and has not been independently verified against an official, reputable ground truth. A dedicated deployment validation step should confirm these addresses after script execution.

3 `TimelockController` v5.4.0 → v5.5.0 changes

Context: OpenZeppelin `TimelockController` v5.4.0 audit

Description: OpenZeppelin's `TimelockController` changes from version v5.4.0 (`f6fea857`) to v5.5.0 were reviewed.

The contract changes are limited to internal variable renaming and a visibility lint change on `updateDelay`, with no side effects for the deployed `TimelockController`. Changes in its dependencies between these versions were also reviewed: `Address.verifyCallResult` was refactored but preserves behavior, and no other meaningful dependency code changes were identified.

4 Basin Scripts

4.1 Low Risk

4.1.1 `SetupJTRSYMorphoUsdtBasin.s.sol` refactor

Severity: Low Risk

Context: SetupJTRSYMorphoUsdtBasin.s.sol#L43

Description: The `SetupJTRSYMorphoUsdtBasin.s.sol` script deviates from the other scripts in several ways.

1. SetupJTRSYMorphoUsdtBasin.s.sol#L43: The setup script deploys a new factory each time instead of reusing the existing factory addresses used by the other scripts. This can lead to duplicate factories that make it hard to track the entire set of Basins.
2. SetupJTRSYMorphoUsdtBasin.s.sol#L22: Its `JTRSY_CHRONICLE_RATE_PROVIDER` does not match the `SetupJTRSYUsdsUsdcBasin.JTRSY_CHRONICLE_RATE_PROVIDER` rate provider. Currently, its JTRSY rate provider is not `toll`ed and will revert.
3. Lack of role setups: The other scripts configure further roles and states that are missing here:
   - `basin.grantRole(basin.MANAGER_ADMIN_ROLE(), Ethereum.GROVE_PROXY);`
   - `basin.grantRole(basin.PAUSER_ROLE(),  Ethereum.ALM_FREEZER);`
   - `basin.grantRole(basin.REDEEMER_ROLE(), issuerRedeemer);`
   - `basin.setFeeBounds(0, 500);`
   - `basin.grantRole(basin.OWNER_ROLE(),  adminTimelock);`
   - `basin.revokeRole(basin.OWNER_ROLE(), deployer);`
   - 4x `basin.setPaused(..);`

Recommendation: Consider refactoring it to match the other scripts more closely by using the same `BasinSetup.sol` library for all core parts of the deployments.

Grove: Fixed in commit 6fdcbd51.

Cantina Managed: Fix verified.

4.2 Informational

4.2.1 `REDEEMER_ROLE` is held by an issuer-controlled address, not by Basin governance

Severity: Informational

Context: script/SetupBUIDLUsdsUsdcBasin.s.sol#L22, script/SetupJTRSYUsdsUsdcBasin.s.sol#L22

Description: `REDEEMER_ROLE` on both production Basins is granted to an issuer-controlled address and not to Basin governance or Grove Proxy:

- **BUIDL** (`SECURITIZE_REDEEMER_ADDRESS`): Fireblocks-managed EOA
- **JTRSY** (`JTRSY_REDEEMER_ADDRESS`): Gnosis Safe 1.4.1, `2 owners` , `threshold = 1`

Only this addresses can call `GroveBasin.initiateRedeem` / `GroveBasin.completeRedeem` .

Recommendations:

1. Document the design choice explicitly so future operators understand the `REDEEMER_ROLE` .
2. Consider communicating to JTRSY to increase the `Safe` threshold to 2-of-2 (or higher).
3. Consider an additional `REDEEMER_ROLE` to `GROVE_PROXY` (or a Sky-controlled multisig) as a fallback so Sky can trigger redemptions if the issuer is unresponsive. `PAUSER_ROLE` can already revoke the issuer's redeemer role in an emergency.

Grove: Acknowledged. This design choice has been documented and communicated to issuers. 3 is unfeasible at the moment given regulatory constraints.

Cantina Managed: Acknowledged.

4.2.2 Third-party post-deployment steps required

Severity: Informational

Context: DeployChronicleRateProvider.s.sol#L68

Description: For the Basin deployments to work in practice, third parties need to grant access to the contracts after deployment, for example:

1. `ChronicleRateProvider` calls `readWithAge()` , which requires the deployed rate provider to be `toll` ed via `oracle.kiss(chronicleRateProvider)` . This must be performed externally after deployment.
2. BUIDL transfer allowlisting is an external dependency.

Recommendation: Document these steps.

Grove: Documented here in the Pre-requirements section.

Cantina Managed: Acknowledged.

4.2.3 Lingering deployer roles

Severity: Informational

Context: DeployTimelockController.s.sol#L49, BasinSetup.sol#L55

Description: The scripts grant privileged roles to the deployer. Some of these roles are not revoked at the end of the script and the deployer still has the rights:

1. On `TimelockController` : Still has `DEFAULT_ADMIN_ROLE` role.
2. On `Basin` : Still has `MANAGER_ADMIN_ROLE` role.

Recommendation: Renounce/revoke all roles of the deployer at the end of the script or verify afterwards that the expected contract setup is as expected.

Grove: For the first two instances, we required a non-atomic deployment. However, we have since revoked the deployer roles.

Cantina Managed: Acknowledged.

4.2.4 Minor issues

Context: (*See each case below*)

Description:

1. DeployFixedRateProvider.s.sol#L16: The script still includes a `run()` helper related to a removed unused `run(uint256)` flow, with the rate hardcoded to `1e27` .

2. BasinSetup.sol#L23: The `seedAmount` can be defined at the top and be reused consistently, including in the balance check.
3. BasinSetup.sol#L19: Replace `msg.sender` with the `deployer` parameter to ensure the intended deployer address is used consistently.
4. BasinSetup.sol#24: Consider using `safeApprove` from `SafeERC20` in BasinSetup like in other contracts.
5. SetupJTRSYMorphoUsdtBasin.s.sol#L27: Consider removing `vm.createSelectFork(getChain("mainnet").rpcUrl)` in `SetupJTRSYMorphoUsdtBasin` all other scripts don't use it. The command overrides the `--rpc-url` passed as parameter when executing the script.
6. DeployChronicleRateProvider.s.sol#L39: For consistency with the other scripts, consider removing the `run` method without a parameter from `DeployChronicleRateProvider`.

Recommendation: Consider addressing the mentioned issues.

Grove: 1 to 4, and 6: acknowledged. 5 has been fixed in commit 6fdcbd51.

Cantina Managed: Verified.

4.2.5 Minor issues, Typos & Documentation

Context: (*See each case below*)

Description:

1. `DeployChronicleRateProvider.s.sol#L39-L45`, `DeployChronicleRateProvider.s.sol#L60-L68`, and `ChronicleRateProvider.sol#L25-L28`: The default `run()` path hardcodes `oracle = address(0)` and then checks it before calling `_deploy`, but `_deploy` already rejects a zero oracle and the `ChronicleRateProvider` constructor also rejects it. Consider removing the no-argument `run()` helper or collapsing the redundant validation.
2. `DeployFixedRateProvider.s.sol#L12-L23`: The usage examples still show `--sig "run(uint256)"`, but the script only exposes `run()` and deploys the default `1e27` rate. The docs should show the actual callable.
3. `DeployFixedRateProvider.s.sol#L34-L44`: The NatSpec says to modify `DEFAULT_RATE`, but no `DEFAULT_RATE` constant exists. The default rate is a local `uint256 rate = 1e27`.
4. `SetupBUIDLUsdsUsdcBasin.s.sol#L6`: `Ethereum` is imported but unused in this script. Remove the import or use the registry constants directly instead of duplicating the addresses.
5. `SetupBUIDLUsdsUsdcBasin.s.sol#L25-L32` and `SetupBUIDLUsdsUsdcBasin.s.sol#L83-L100`: `deploy()` creates and registers a `BUIDLTokenRedeemer`, but `run()` only returns/logs the basin and pocket. This differs from the JTRSY script, which returns/logs the token redeemer. The BUIDL script should return and log `tokenRedeemer` so operators do not need to recover it from traces or logs.
6. `SetupBUIDLUsdsUsdcBasin.s.sol#L16`, `SetupJTRSYUsdsUsdcBasin.s.sol#L16`, and `SetupJTRSYMorphoUsdtBasin.s.sol#L22-L23`: The fixed 1:1 rate provider at `0x7928A185B8137D1CD2a0996a810A04dB2837419D` is documented as a `ChronicleRateProvider`, but it is a `FixedRateProvider`. Update the comments to avoid implying Chronicle oracle behavior where the provider is fixed.

Recommendation: Consider addressing the issues mentioned above.

Sky: Acknowledged.

Cantina Managed: Acknowledged.