Sky: LZ Solana Governance
Security Review

Cantina Managed review by:

Christoph Michel, Lead Security Researcher
Mario.eth, Lead Security Researcher

October 28, 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

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

From Sep 22nd to Sep 26th the Cantina team conducted a review of sky-oapp-oft on commit hash efc7a928. 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	0	0	0
Gas Optimizations	0	0	0
Informational	6	5	1
Total	**6**	**5**	**1**

The Cantina Managed team reviewed Sky's `sky-oapp-oft` holistically on commit hash c9db37d8 and concluded that all findings were addressed and no new vulnerabilities were identified.

3 Findings

3.1 Informational

3.1.1 `LzReceiveTypesV2Accounts` documentation

Severity: Informational

Context: lz_receive_types_v2.rs#L13-L37

Description: The documentation about the `LzReceiveTypesV2Accounts` struct states the following:

```
/// The payload returned from `lz_receive_types_info` when version == 2.
/// Provides information needed to construct the call to `lz_receive_types_v2`.
///
/// This structure is stored at a deterministic PDA and serves as the bridge between
/// the version discovery phase and the actual execution planning phase of V2.
///
/// # Execution Flow
/// 1. **Version Discovery**: The Executor calls `lz_receive_types_info` using the PDA. If the
///    version is 2, it decodes the returned payload into LzReceiveTypesV2Accounts.
/// 2. **Execution Planning**: The Executor constructs the accounts from LzReceiveTypesV2Accounts to
///    call `lz_receive_types_v2`, which returns the complete execution plan.
///
/// # Storage Location
/// This data is stored at a PDA derived using:
/// ```
/// seeds = [LZ_RECEIVE_TYPES_SEED, &oapp_address.to_bytes()]
/// ```
///
/// # Fields
/// - `accounts`: A vector of `Pubkey` containing the static accounts needed to construct the
///   `lz_receive_types_v2` instruction.
#[derive(AnchorSerialize, AnchorDeserialize, Clone)]
pub struct LzReceiveTypesV2Accounts {
    pub accounts: Vec<Pubkey>,
}
```

However, this part is not how `lz_receive_types` v2 works in practice:

> This structure is stored at a deterministic PDA
> /// # Storage Location
> /// This data is stored at a PDA derived using:
> /// ```
> /// seeds = [LZ_RECEIVE_TYPES_SEED, &oapp_address.to_bytes()]
> /// ```
> ///

This structure is not the structure stored at this deterministic PDA, it is the "return value" of the `lz_receive_types_info` instruction.

For v1, there is a similar structure stored in a PDA with these seeds that will be read and deserialized and its `accounts` will be provided as the context for the `lz_receive_types` call.

For v2, the PDA from this seed (which is a different struct than the `LzReceiveTypesV2Accounts`) will be provided as context to `lz_receive_types_info` but the `lz_receive_types_v2` instruction context will fully depend on what is returned from the `info` instruction, not what is stored in this PDA. The `LzReceiveTypesV2Accounts` struct is never directly serialized into a PDA.

Recommendation: Consider updating the documentation to reflect the v2 behavior.

Sky: PR 169 updates the documentation.

Cantina Managed: Fix verified.

3.1.2 Not all instruction from `lz_receive_types_v2` are required to be executed

Severity: Informational

Context: lz_receive_types_v2.rs#L56

Description: The `lz_receive_types_v2` instruction returns a vector of `Instruction`s that should be executed. It is expected to contain one `lz_receive` call and any amount of other instructions.

However, as execution is permissionless, anyone can attempt execution. A malicious executor can omit all other instructions besides `lz_receive`. The `lz_receive` execution will clear the x-chain message and make it unreplayable. The other instructions will never be executed. The `lz_receive` instruction itself could have relied on the previous encoded instructions to set up proper state for it. Skipping this setup could lead to unexpected executions in `lz_receive`.

Recommendation: As with LZ options, additional instructions returned by `lz_receive_types_v2` are not enforced and transaction introspection must be used if one relies on them. Consider documenting this behavior as it might not be intuitive for integrators.

Sky: LayerZero shall include comments describing the lifecycle of the OApp.

Cantina Managed: Acknowledged.

3.1.3 `impl From<Instruction> for GovernanceMessage` is not used

Severity: Informational

Context: msg_codec.rs#L158-L175

Description: The `impl From<Instruction> for GovernanceMessage` trait implementation is currently not used and unlikely to be used in the future. The relevant conversion is from a `GovernanceMessage` to an `Instruction`.

Recommendation: Consider removing this code.

Sky: Fixed in commit 6565a09a.

Cantina Managed: Fix verified.

3.1.4 Update LayerZero dependency to latest `lz_receive_types_v2` changes

Severity: Informational

Context: Cargo.toml#L23-L33

Description: The LayerZero `lz_receive_types_v2` changes required for the Sky Governance program are implemented in PR 152. The current dependency used by `Cargo.toml` points to an outdated, deleted commit.

Recommendation: Consider updating the dependency to the latest version after `lz_receive_types_v2` have been finalized.

Sky: Fixed in commit 2ff4988f.

Cantina Managed: Fix verified.

3.1.5 Having the Sky Governance OApp creation permissionless is risky

Severity: Informational

Context: instructions/init_governance.rs#L8-L26

Description: `init_governance` is currently permissionless. Any wallet can call it to create a new Governance OApp for any `id`, choose an arbitrary `admin` pubkey, and (as part of the same instruction) register that admin as the OApp in the Endpoint via the `register_oapp` CPI.

This could cause front-running issues as well as registering Governance OApp by anyone in a Sky controlled program might not be the right approach.

Recommendation: Consider gating the init to be the deployer of the program.

Sky: Fixed in commit 5eda2284.

Cantina Managed: Fix verified.

3.1.6 Missing tests

Severity: Informational

Context: See below

Description: The `init_governance` flow is missing a test to highlight the actual setup of the Sky Governance Oapp.

The flow should be:

- Deploy Sky Governance app with a local admin.
- Configure the `remote` and `SetOAppConfig`.
- Transfer the admin to the `cpi_authority`.
- Call again `SetOAppConfig` and make sure we do not hit the CPI depth limit.

This test highlights how the Sky Governance OApp will be setup and makes sure we do not hit the CPI depth limit as `cpi_authority` will be the admin.

Recommendation: Consider implementing the tests mentioned above.

Sky: The requested automated test scenario has been added and is passing. See PR 28.

Cantina Managed: The mentioned tests for the scenario described in the issue are in `test/solana/suites/governance.test.ts` (of course, with hardcoded messages that get delivered, not fully EVM-to-Solana end-to-end, but that's reasonable and fine for this part).