Drips contracts

Security Review

Cantina Managed review by:
Deadrosesxyz, Lead Security Researcher
Akshay Srivastav, Security Researcher
High Byte, Security Researcher

September 21, 2024

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	Description
Critical	*Must* fix as soon as possible (if already deployed).
High	Leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
Medium	Global losses <10% or losses to only a subset of users, but still unacceptable.
Low	Losses will be annoying but bearable. Applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.
Gas Optimization	Suggestions around gas saving practices.
Informational	Suggestions around best practices or readability.

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 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

Drips is a protocol and app built on Ethereum that enables organizations and individuals to directly and publicly provide funding to the free and open source software projects they depend on the most.

Drips also includes gas-optimized and integrated primitives for streaming and splitting tokens, allowing users and web3 apps to stream and split funds by the second with continuous settlement for use cases like contributor payments, vesting and subscription memberships.

From Aug 31st to Sep 4th the Cantina team conducted a review of drips-contracts on commit hash 3ab49b25. The team identified a total of **5** issues in the following risk categories:

- Critical Risk: 0
- High Risk: 0
- Medium Risk: 1
- Low Risk: 3
- Gas Optimizations: 0
- Informational: 1

3 Findings

3.1 Medium Risk

3.1.1 The `AxelarBridgedGovernor` contract cannot support non-zero `value` cross-chain calls

Severity: Medium Risk

Context: BridgedGovernor.sol#L183-L188

Description: The `AxelarBridgedGovernor` contract lacks `receive` & `fallback` functions due to which there is no easy way to send ETH to `AxelarBridgedGovernor` contract. Moreover the `execute` function is not `payable`.

Combination of the above two behaviours results in `AxelarBridgedGovernor`'s inability to support `Calls` with non-zero `Call.value`.

Recommendation: Consider adding `receive` function to `AxelarBridgedGovernor` contract.

Drips: Fixed in commit f3d7e6f7.

Cantina Managed: Fixed.

3.2 Low Risk

3.2.1 `AxelarBridgedGovernor` allows for out-of-order execution in a specific edge case

Severity: Low Risk

Context: BridgedGovernor.sol#L50

Description: After a message is successfully sent from the `_ownerChain`, anyone can call `AxelarBridgedGovernor`'s `execute` function on the destination chain and execute the calls.

However, since the `execute` function lacks a `nonReentrant` guard, if there are 2 scheduled messages waiting to be executed, a call from the first one could invoke execution of the next one, before the first one fully finishes.

Because of this, messages might be executed in an order which the DAO might not have intended.

Recommendation: Add a `nonReentrant` guard to `execute`

Since LayerZero's `EndpointV2.lzReceive` is a public function, the same out-of-order execution can also be performed for `LZBridgedGovernor.lzReceive` calls. Hence the `nonReentrant` guard should also be added to `LZBridgedGovernor.lzReceive` function.

Drips: Fixed in commit f3d7e6f7.

Cantina Managed: The changes in commit f3d7e6f7 for this issue look good.

3.2.2 `BridgedGovernor`s cannot transfer chain's native tokens to EOAs

Severity: Low Risk

Context: BridgedGovernor.sol#L139, BridgedGovernor.sol#L31-L36

Description: The `runCalls` function uses `Address.functionCallWithValue` function to execute a `Call` which reverts in case the `Call.target` is an EOA. This leads to inability of BridgedGovernors in sending native tokens directly to an EOA.

As `LZBridgedGovernor.lzReceive` allows `msg.value >= message.value`, there could be a need for the BridgedGovernor to simply send its extra ETH balance to an EOA which won't be possible in the current code implementation.

Recommendation: Consider implementing the ability to send native tokens to EOAs.

Drips: Fixed in commit f3d7e6f7.

Cantina Managed: Please also remove the comment above `executeCalls` function which says:

```
/// All the targets must be smart contracts, calling an EOA will revert.
```

3.2.3 The `constructor` of `BridgedGovernorProxy` cannot support non-zero `value` calls

Severity: Low Risk

Context: BridgedGovernor.sol#L205

Description: The `constructor` of `BridgedGovernorProxy` contract executes `Calls`. Since a `Call` can have non-zero `value` field, it is better to mark the `constructor` as `payable`.

Recommendation: Add this change:

```
- constructor(address logic, Call[] memory calls) ERC1967Proxy(logic, "") {
+ constructor(address logic, Call[] memory calls) ERC1967Proxy(logic, "") payable {
      runCalls(calls);
  }
```

Drips: Fixed in commit f3d7e6f7.

Cantina Managed: Fixed.

3.3 Informational

3.3.1 External functions of `AxelarBridgedGovernor` contract are missing `onlyProxy` modifier

Severity: Informational

Context: BridgedGovernor.sol#L155, BridgedGovernor.sol#L183-L188

Description: All the external functions of `LZBridgedGovernor` have `onlyProxy` modifier which prevents direct access of those functions on implementation contract.

However the `execute` and `ownerChain` functions of `AxelarBridgedGovernor` contract do not have the `onlyProxy` modifier. This breaks the code consistency of governor implementations.

Recommendation: Consider adding `onlyProxy` modifier to `execute` and `ownerChain` functions.

Drips: Fixed in commit f3d7e6f7.

Cantina Managed: Fixed.