Security Audit Report

Axelar solana-gas-service (3)

Solana program

by ARDA

on April 15, 2026

Table of Contents

Disclaimer **3**

Terminology **4**

Objective **4**

Summary **5**

Inherent Risks **6**

Code Issues & Recommendations **7**

C1: Wrong rent-exemption check may cause treasury garbage collection 7

Disclaimer

The report makes no statements or warranties, either expressed or implied, regarding the security of the code, the information herein or its usage. It also cannot be considered as a sufficient assessment regarding the utility, safety and bugfree status of the code, or any other statements.

This report does not constitute legal or investment advice. It is for informational purposes only and is provided on an "as-is" basis. You acknowledge that any use of this report and the information contained herein is at your own risk. The authors of this report shall not be liable to you or any third parties for any acts or omissions undertaken by you or any third parties based on the information contained herein.

Terminology

Code: The code with which users interact.

Inherent risk: A risk for users that comes from a behavior inherent to the code's design.

Inherent risks only represent the risks inherent to the code's design, which are a subset of all the possible risks. **No inherent risk doesn't mean no risk.** It only means that no risk inherent to the code's design has been identified. Other kind of risks could still be present. For example, the issues not fixed incur risks for the users, or the upgradability of the code might also incur risks for the users.

Issue: A behavior unexpected by the users or by the project, or a practice that increases the chances of unexpected behaviors to appear.

Critical issue: An issue intolerable for the users or the project, that must be addressed.

Major issue: An issue undesirable for the users or the project, that we strongly recommend to address.

Medium issue: An issue uncomfortable for the users or the project, that we recommend to address.

Minor issue: An issue imperceptible for the users or the project, that we advise to address for the overall project security.

Objective

Our objective is to share everything we have found that would help assessing and improving the safety of the code:

1. The **inherent risks** of the code, labelled R1, R2, etc.
2. The **issues** in the **code**, labelled C1, C2, etc.
3. The **issues** in the **testing** of the code, labelled T1, T2, etc.
4. The **issues** in the **other** parts related to the code, labelled O1, O2, etc.
5. The **recommendations** to address each issue.

Summary

Blockchain: Solana

Initial scope

- **Repository:** https://github.com/axelarnetwork/axelar-amplifier-solana
- **Commit:** `30e4195a839ded2d7a695b2e9e71e06242a43b58`
- **Path(s):** `./programs/solana-axelar-gas-service`

Final scope

- **Repository:** https://github.com/axelarnetwork/axelar-amplifier-solana
- **Commit:** `3e7a6f81ea41bc1eafb78dd10acbdbb589bd1a00`
- **Path(s):** `./programs/solana-axelar-gas-service`

1 inherent risk in the final scope

0 issue in the final scope

1 issue reported in the initial scope and 0 remaining in the final scope.

Severity	Reported			Remaining		
	Code	Test	Other	Code	Test	Other
Critical	0	0	0	0	0	0
Major	1	0	0	0	0	0
Medium	0	0	0	0	0	0
Minor	0	0	0	0	0	0

Inherent Risks

R1: Users might be refunded incorrectly, or too late, or even not refunded at all for the excess of tokens they provided to cover gas costs.

This is because the tokens deposited in the Gas Service program are fully managed by trusted operators, who might not perform the refunding as expected in the following situations:

- The operators could make mistakes, e.g. could refund a wrong address, a wrong amount, or a wrong token.
- The operators could be manipulated: anyone with access to an operator's private key can send all the funds deposited in the Gas Service to any address.
- The operators might be inactive, e.g. might have lost access to private keys or might be unavailable, in which case refunds will not be processed.

Code Issues & Recommendations

C1: Wrong rent-exemption check may cause treasury garbage collection

Severity: Major **Status:** Fixed

Description

In the functions `collect_native_fees` and `refund_native_fees`, after transferring lamports out of the treasury, the code checks that the treasury remains rent-exempt:

```
if !Rent::is_exempt(treasury.get_lamports(), Treasury::INIT_SPACE) {
    return Err(ProgramError::InvalidAccountData.into());
}
```

Here, `Treasury::INIT_SPACE` only accounts for the struct data size (1 byte for the `bump` field). However, the actual on-chain account size also includes the 8-byte Anchor discriminator, making the total data length 9 bytes. Since `Rent::is_exempt` computes the minimum lamports required for a given data size, passing only 1 byte instead of 9 bytes results in an underestimated rent-exemption threshold.

Therefore, after the function call, the treasury could end up below the rent-exemption threshold, and then the Solana runtime would garbage-collect the treasury account. In turn, all subsequent Gas Service transactions would fail (e.g., ITS integrations that rely on the Gas Service) until an operator manually calls the `initialize` function to re-create the treasury.

Recommendation

We recommend using `Treasury::DISCRIMINATOR.len() + Treasury::INIT_SPACE` (accounting for the Anchor discriminator) in the rent-exemption check, both in `collect_native_fees` and `refund_native_fees`.

Further, we recommend adding a unit test witnessing that, when these two methods leave the `Treasury` PDA exactly at the rent-exemption threshold, the `Treasury` PDA is not deleted by the runtime at the end of the transaction.