# Security Assessment Report

# Boring Bridge

November 2024

*Prepared for*
**Veda**

certora

# Table of content

**Project Summary** ..... **3**
Project Scope ..... 3
Project Overview ..... 3
Findings Summary ..... 4
Severity Matrix ..... 4
**Detailed Findings** ..... **5**
Low Severity Issues ..... 6
L-01. Program authority and program owner are the same ..... 6
Informational Severity Issues ..... 7
I-01. Upgrading program ownership might deviate implementation from documented specifications ..... 7
I-02. Incorrect assumptions made in documentation ..... 7
I-03. Unnecessary hard coding of space size ..... 8
I-04. Incorrect test setup in documentation ..... 8
I-05. Centralization risks ..... 8
**About Certora** ..... **9**

certora

# Project Summary

## Project Scope

| Project Name | Repository (link) | Latest Commit Hash | Platform |
|---|---|---|---|
| Passkey Module | https://github.com/Veda-Labs/boring-bridge-holder/tree/main | 5239c75 | SVM/Solana |

## Project Overview

This document describes the specification and verification of **Boring Bridge Holder** using manual code review findings. The work was undertaken from **November 5, 2024** to **November 12, 2024**.

The following contract list is included in our scope:

`programs/*`

The team performed a manual audit of all the Solana contracts**.** During the manual audit, the Certora team discovered bugs in the Solana contracts code, as listed on the following page.

certora

## Findings Summary

The table below summarizes the findings of the review, including type and severity details.

| Severity | Discovered | Confirmed | Fixed |
|---|---|---|---|
| Critical | 0 | | |
| High | 0 | | |
| Medium | 0 | | |
| Low | 1 | | |
| Informational | 5 | | |
| **Total** | **6** | | |

## Severity Matrix

| | | | | |
|---|---|---|---|---|
| **Impact** | High | Medium | High | Critical |
| | Medium | Low | Medium | High |
| | Low | Low | Low | Medium |
| | | Low | Medium | High |
| | | **Likelihood** | | |

certora

# Detailed Findings

| ID | Title | Severity | Status |
|---|---|---|---|
| L-01 | Program authority and program owner are the same | Low | Won't Fix |

## Low Severity Issues

### L-01. Program authority and program owner are the same

**Description**:

For full transparency before we dive into the issue: here “program owner” refers to the account who is able to call permissioned functions, but is not able to upgrade the program data directly like the “program authority” can.

According to the boring bridge holder documentation: “

- Program upgrade authority is managed by a Squads multisig

- Program account ownership is managed by the same Squads multisig “

Having both accounts be managed by the same multisig creates a single point of failure. We recommend having a separate multisig for the program account ownership. This would further enhance the operational security of the protocol by potentially limiting the harm done if the program owner's private key is compromised.

**Customer’s response: Acknowledged, but wont’ fix.**

## Informational Severity Issues

### I-01. Upgrading program ownership might deviate implementation from documented specifications

**Description**:

According to the documentation provided, the program authority and the program owner are supposed to be managed by the same multisig. Through ownership transfer a program owner owned by a different multisig could gain ownership, making this specification untrue as there are no checks in the program ensuring otherwise.

As explained in the previous issue, we believe this specification increases single-point-of-failure risks, and therefore, recommend not enforcing it.

**Customer's response:** Acknowledged.

### I-02. Incorrect assumptions made in documentation.

**Description**: As discussed with the team "This design choice is necessary since PDAs with data cannot transfer lamports" is not necessarily true as the PDA owner can transfer lamports on behalf of the PDA. However, we recommend keeping the current architecture of the strategist account as it is simpler and decreases single-point-of-failure risks.

As requested by the team, here are resources that can be used for future reference regarding transferring lamports out of a PDA:

- https://docs.rs/anchor-lang/latest/anchor_lang/trait.Lamports.html#method.sub_lamports
- https://solana.com/docs/core/cpi#how-to-write-a-cpi
- https://www.rareskills.io/post/solana-account-owner Section: Transferring SOL out of a PDA: Crowdfund example

**Customer's response:** Fixed at https://github.com/Veda-Labs/boring-bridge-holder/pull/5/commits/65cb71dcfe0e3e42551b7c80bb4105c43638fd57

## I-03. Unnecessary hard coding of space size

**Description**: Hard coding the 'space' elements for PDAs is prone to manual errors. We recommend instead using 'size_of' , which mitigates manual errors and leads to a more pleasant development experience by calculating the necessary space to be allocated whenever necessary.

Implementation example resource at the **Account initialization boilerplate code** section of https://www.rareskills.io/post/solana-initialize-account .

**Customer's response:** Fixed at https://github.com/Veda-Labs/boring-bridge-holder/pull/5/commits/527a86463a0538382069158cf44330d606f56f96

## I-04. Incorrect test setup in documentation

**Description**: Documentation regarding running of tests is incorrect. Starting a local Solana test validator via 'solana-test-validator' is unnecessary. This step can be removed as 'anchor test' starts its own local validator by default, and will clash with the previous, unless '--skip-local-validator' flag is passed.

**Customer's response:** Fixed at https://github.com/Veda-Labs/boring-bridge-holder/pull/5/commits/f364c572a25c9eb1449c789cc86683e1276c2974

## I-05. Centralization risks

**Description**: Program authority, program owner and strategist are all fully under control of the team and can ultimately be used to send funds to any arbitrary address at any point in time. These trust assumptions are equivalent to the trust assumptions on the EVM side of the bridge and are mitigated through the use of multisigs to control said accounts.

**Customer's response:** Acknowledge.

# About Certora

Certora is a Web3 security company that provides industry-leading formal verification tools and smart contract audits. Certora's flagship security product, Certora Prover, is a unique SaaS product that automatically locates even the most rare & hard-to-find bugs on your smart contracts or mathematically proves their absence. The Certora Prover plugs into your standard deployment pipeline. It is helpful for smart contract developers and security researchers during auditing and bug bounties.

Certora also provides services such as auditing, formal verification projects, and incident response.