Parabol Labs, Protocol Contracts

Security Assessment (Summary Report)

January 24, 2025

Prepared for:
Emre Colakoglu
Parabol Labs

Prepared by: **Tarun Bansal**

Table of Contents

Project Summary

Contact Information

The following project manager was associated with this project:

Jeff Braswell, Project Manager
jeff.braswell@trailofbits.com

The following engineering director was associated with this project:

Josselin Feist, Engineering Director, Blockchain
josselin.feist@trailofbits.com

The following consultant was associated with this project:

Tarun Bansal, Consultant
tarun.bansal@trailofbits.com

Project Timeline

The significant events and milestones of the project are listed below.

Date	Event
December 23, 2024	Pre-project kickoff call
December 30, 2024	Delivery of report draft
January 6, 2025	Report readout meeting
January 24, 2025	Delivery of final summary report

Project Targets

The engagement involved a review and testing of the following target.

parabol-protocol-contracts

Repository	https://github.com/Parabol-Finance/parabol-protocol-contracts
Version	Diff between `03ee167` and `5b046f6`
Type	EVM
Platform	Solidity

Executive Summary

Engagement Overview

Parabol Labs engaged Trail of Bits to review the security of the Parabol Protocol smart contract updates. These updates add new features that allow third parties to integrate with the Parabol Protocol to provide yield to their customers.

One consultant conducted the review from December 26 to December 27, 2024, for a total of two engineer-days of effort. With full access to source code and documentation, we performed static and dynamic testing of the target codebase, using automated and manual processes.

Observations and Impact

The Parabol Labs team updated the Parabol Protocol contracts to add new features that facilitate institutional partner integration and batch operations. These features include a new partner manager contract and a batch claim function, among other changes. We reviewed the changes in the following smart contracts:

- `ParabolUSD.sol`
- `NonFungibleNotePosition.sol`
- `ReserveStabilityPool.sol`
- `NoncesUpgradeable.sol`
- `ERC20AuthUpgradeable.sol`
- `ERC20BaseUpgradeable.sol`
- `ERC721PermitUpgradeable.sol`
- `PartnerFeeManagerUpgradeable.sol`
- `FeedSignatureVerifierUpgradeable.sol`
- `INonFungibleNotePosition.sol`
- `IReserveStabilityPool.sol`

Due to the time-boxed nature of this review, we did not perform a comprehensive review of the existing unchanged functionality of the above-listed smart contracts.

The changes did not introduce any high-severity security vulnerabilities into the codebase. We discovered an informational-severity issue related to an incorrect argument in the

`Approval` event emitted from the `NonFungibleNotePosition` contract (TOB-PRBLDIFF-1), affecting the off-chain data indexers.

Recommendations

Remediate the findings disclosed in this report and improve the test suite to check events emitted by transactions.

Summary of Findings

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

ID	Title	Type	Severity
1	Incorrect argument in Approval event emitted from NonFungibleNotePosition contract	Auditing and Logging	**Informational**

Detailed Findings

1. Incorrect argument in Approval event emitted from NonFungibleNotePosition contract

Severity: **Informational**	Difficulty: **Low**
Type: Auditing and Logging	Finding ID: TOB-PRBLDIFF-1
Target: `contracts/base/ERC721PermitUpgradeable.sol`	

Description

The `NonFungibleNotePosition` contract extends the `ERC721PermitUpgradable` contract. The `ERC721PermitUpgradable` contract overrides the `_update` function of the `ERC721Upgradable` contract to emit an `Approval` event. This `Approval` event is required to update the approval state in the off-chain data indexer.

However, the first argument of the `Approval` event is the `auth` variable instead of the `from` variable. This causes the wrong approval state in the off-chain indexer to be cleared or the event to be ignored if the approval from the `auth` value does not exist for the `tokenId` value in the indexer database.

```
function _update(
    address to,
    uint256 tokenId,
    address auth
) internal virtual override returns (address) {
    address from = ERC721Upgradeable._update(to, tokenId, auth);
    if (from != address(0)) {
        emit Approval(auth, address(0), tokenId);
    }
    return from;
}
```

Figure 1.1: contracts/base/ERC721PermitUpgradeable.sol#L155-L165

Recommendations

Short term, use the `from` variable as the first argument for the `Approval` event.

Long term, improve the test suite to check events emitted by transactions to ensure they are correct.

A. Vulnerability Categories

The following tables describe the vulnerability categories, severity levels, and difficulty levels used in this document.

Vulnerability Categories	
Category	**Description**
Access Controls	Insufficient authorization or assessment of rights
Auditing and Logging	Insufficient auditing of actions or logging of problems
Authentication	Improper identification of users
Configuration	Misconfigured servers, devices, or software components
Cryptography	A breach of system confidentiality or integrity
Data Exposure	Exposure of sensitive information
Data Validation	Improper reliance on the structure or values of data
Denial of Service	A system failure with an availability impact
Error Reporting	Insecure or insufficient reporting of error conditions
Patching	Use of an outdated software package or library
Session Management	Improper identification of authenticated users
Testing	Insufficient test methodology or test coverage
Timing	Race conditions or other order-of-operations flaws
Undefined Behavior	Undefined behavior triggered within the system

Severity Levels	
Severity	**Description**
Informational	The issue does not pose an immediate risk but is relevant to security best practices.
Undetermined	The extent of the risk was not determined during this engagement.
Low	The risk is small or is not one the client has indicated is important.
Medium	User information is at risk; exploitation could pose reputational, legal, or moderate financial risks.
High	The flaw could affect numerous users and have serious reputational, legal, or financial implications.

Difficulty Levels	
Difficulty	**Description**
Undetermined	The difficulty of exploitation was not determined during this engagement.
Low	The flaw is well known; public tools for its exploitation exist or can be scripted.
Medium	An attacker must write an exploit or will need in-depth knowledge of the system.
High	An attacker must have privileged access to the system, may need to know complex technical details, or must discover other weaknesses to exploit this issue.

About Trail of Bits

Founded in 2012 and headquartered in New York, Trail of Bits provides technical security assessment and advisory services to some of the world's most targeted organizations. We combine high-end security research with a real-world attacker mentality to reduce risk and fortify code. With 100+ employees around the globe, we've helped secure critical software elements that support billions of end users, including Kubernetes and the Linux kernel.

We maintain an exhaustive list of publications at https://github.com/trailofbits/publications, with links to papers, presentations, public audit reports, and podcast appearances.

In recent years, Trail of Bits consultants have showcased cutting-edge research through presentations at CanSecWest, HCSS, Devcon, Empire Hacking, GrrCon, LangSec, NorthSec, the O'Reilly Security Conference, PyCon, REcon, Security BSides, and SummerCon.

We specialize in software testing and code review projects, supporting client organizations in the technology, defense, and finance industries and government entities. Notable clients include HashiCorp, Google, Microsoft, Western Digital, and Zoom.

Trail of Bits also operates a center of excellence with regard to blockchain security. Notable projects include audits of Algorand, Bitcoin SV, Chainlink, Compound, Ethereum 2.0, MakerDAO, Matic, Uniswap, Web3, and Zcash.

To keep up to date with our latest news and announcements, please follow @trailofbits on Twitter and explore our public repositories at https://github.com/trailofbits. To engage us directly, visit our "Contact" page at https://www.trailofbits.com/contact or email us at info@trailofbits.com.

Trail of Bits, Inc.
497 Carroll St., Space 71, Seventh Floor
Brooklyn, NY 11215
https://www.trailofbits.com
info@trailofbits.com

Notices and Remarks

Copyright and Distribution

Test Coverage Disclaimer

All activities undertaken by Trail of Bits in association with this project were performed in accordance with a statement of work and agreed upon project plan.

Security assessment projects are time-boxed and often reliant on information that may be provided by a client, its affiliates, or its partners. As a result, the findings documented in this report should not be considered a comprehensive list of security issues, flaws, or defects in the target system or codebase.

Trail of Bits uses automated testing techniques to rapidly test the controls and security properties of software. These techniques augment our manual security review work, but each has its limitations: for example, a tool may not generate a random edge case that violates a property or may not fully complete its analysis during the allotted time. Their use is also limited by the time and resource constraints of a project.