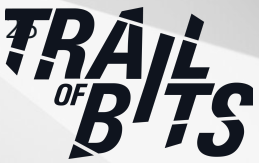

Puffer Finance Contracts

Security Assessment (Summary Report)

March 25, 2024

Prepared for:
Jason Vranek
Puffer Finance

Prepared by: **Justin Jacob**

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, as well as 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.
228 Park Ave S #80688
New York, NY 10003
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.

Table of Contents

Project Summary

Contact Information

The following project manager was associated with this project:

Mary O'Brien, Project Manager
mary.obrien@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:

Justin Jacob, Consultant
justin.jacob@trailofbits.com

Project Timeline

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

Date	Event
February 22, 2024	Pre-project kickoff call
March 5, 2024	Delivery of report draft
March 5, 2024	Report readout meeting
March 25, 2024	Delivery of summary report

Project Targets

The engagement involved a review and testing of the targets listed below.

Puffer Pool: ValidatorTicket, PufferOracle

Repository	https://github.com/PufferFinance/PufferPool/
Version	`95eb5129f032d2bc60f6e068edc94e36dda8faf9`
Type	Solidity
Platform	EVM

pufETH: PufferDepositorV2, PufferVaultV2

Repository	https://github.com/PufferFinance/pufETH/
Version	`6dd2b644030313f7393e68fd715759259d63e670`
Type	Solidity
Platform	EVM

Executive Summary

Engagement Overview

Puffer Finance engaged Trail of Bits to review the security of its `pufETH` and `PufferPool` repositories. In the `pufETH` repository, two files were in scope: `PufferVaultV2.sol` and `PufferDepositorV2.sol`. The `PufferVaultV2` contract is a 4626-compliant vault implementation that allows users to deposit ETH, wETH, and stETH, while the `PufferDepositorV2` contract allows users to deposit stETH and wstETH into the vault. In the `PufferPool` repository, the files `PufferOracle.sol` and `ValidatorTicket.sol` were in scope. `ValidatorTicket` represents an ERC-20 token used as collateral for funding validators, while `PufferOracle` is intended to calculate the price of `ValidatorTicket`.

One consultant conducted the review from February 26 to March 4, 2024, for a total of six engineer-days of effort. With full access to source code and documentation, we performed static and dynamic testing of the project targets, using automated and manual processes.

Observations and Impact

Overall, the codebase is easy to understand and implements components based on reusable libraries. We focused on validating the vault's behavior and 4626 conformance, investigating whether deposits and withdrawals can be prevented or stolen, and examining edge cases in integrating with stETH. Furthermore, we investigated the behavior and purchase of `ValidatorTicket` tokens, as well as the general behavior of the `PufferOracle` contract. We wrote a fuzzing harness to test some ERC-4626 properties, but we recommend further fuzzing—in particular, integrating the three different types of collateral tokens.

Recommendations

Based on the findings and codebase maturity evaluation in this report, we recommend that Puffer Finance take the following steps:

- **Remediate the findings disclosed in this report.** These findings should be addressed as part of a direct remediation or as part of any refactor that may occur when addressing other recommendations.
- **Expand the fuzz testing suite.** The current fuzz testing suite is a good baseline, but further testing, especially tailored to protocol-specific invariants, will help uncover edge cases. Guidance on introducing stateful fuzzing can be found in Trail of Bits' Learn how to fuzz like a pro series on YouTube.

Codebase Maturity Evaluation

Trail of Bits uses a traffic-light protocol to provide each client with a clear understanding of the areas in which its codebase is mature, immature, or underdeveloped. Deficiencies identified here often stem from root causes within the software development life cycle that should be addressed through standardization measures (e.g., the use of common libraries, functions, or frameworks) or training and awareness programs.

Category	Summary	Result
Arithmetic	The codebase does not involve any complex mathematical operations and instead uses floating point division and multiplication. The risks of overflows are mitigated due to the use of Solidity ^0.8.0. Care must be taken to ensure the correctness and precision of these operations, as any unintended rounding can lead to edge cases in the contracts. We recommend further fuzz testing the vault with Echidna and `medusa`.	**Satisfactory**
Auditing	We found one instance of state changes not being followed by event emission (TOB-PUF-6). While the Puffer team has said it will partner with BlockSec and Hexagate for monitoring, we did not receive detailed documentation regarding the scope and coverage of this monitoring.	**Moderate**
Authentication / Access Controls	The system has various permissions for different functions and adequate NatSpec comments explaining who has the permissions to call each function. However, the contracts use the restricted modifier as the access control for every role, making the powers and privileges of the contract opaque to end users. The access controls should be separated in a granular way, with detailed documentation explaining the powers and privileges of each role in the system.	**Satisfactory**
Complexity Management	The codebase's various components are well separated, and the contracts' call sequences are easy to understand. Every function has NatSpec or inline comments explaining its purpose, and the functions are simple to analyze. However, the protocol may have to account for edge cases due to support for three different types of	**Moderate**

	collateral and due to timing concerns about withdrawals because of how the vault works when validators are provisioned.	
Decentralization	The decentralized autonomous organization (DAO) is responsible for setting system parameters; however, it is unclear how the DAO operates and when it votes. It is also unclear when powers from the admin team will be transferred to the DAO. Additionally, the admin team can set withdrawal limits and update them on the vault and have tremendous impact over the system in general. Updating users on the process of decentralization will help garner trust in the system.	**Moderate**
Documentation	The system is lacking documentation about access controls and the DAO's powers and privileges. However, there is good documentation about the codebase in inline comments. There is also high-level protocol documentation about various parts of the system. Expanding on this documentation and further describing various call sequences and user flows with diagrams would be beneficial.	**Moderate**
Low-Level Manipulation	The codebase makes extremely limited use of low-level manipulation.	**Satisfactory**
Testing and Verification	Most of the issues we found would have been caught with a larger testing suite. In addition, the protocol lacks fuzz or unit testing that encompasses a wide range of scenarios. We recommend expanding the implemented fuzz testing suite with Puffer Vault–specific invariants.	**Moderate**
Transaction Ordering	We found two instances where transaction reordering could lead to users performing arbitrage or causing a denial of service. Long-term investigation is required to determine the robustness of the protocol to more intricate instances of transaction reordering.	**Moderate**

Summary of Findings

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

ID	Title	Type	Severity
1	An attacker can cause a DOS for withdrawals	Denial of Service	High
2	setWithdrawalLimit does not reset assetsWithdrawnToday	Data Validation	Medium
3	stETH rebasing creates arbitrage opportunities	Timing	Low
4	Vault withdrawals and redemptions can revert	Timing	Medium
5	ValidatorTicket purchase can revert due to strict equality	Data Validation	Informational
6	updateDailyWithdrawals does not emit an event	Auditing and Logging	Low

A. Code Maturity Categories

The following tables describe the code maturity categories and rating criteria used in this document.

Code Maturity Categories	
Category	**Description**
Arithmetic	The proper use of mathematical operations and semantics
Auditing	The use of event auditing and logging to support monitoring
Authentication / Access Controls	The use of robust access controls to handle identification and authorization and to ensure safe interactions with the system
Complexity Management	The presence of clear structures designed to manage system complexity, including the separation of system logic into clearly defined functions
Cryptography and Key Management	The safe use of cryptographic primitives and functions, along with the presence of robust mechanisms for key generation and distribution
Decentralization	The presence of a decentralized governance structure for mitigating insider threats and managing risks posed by contract upgrades
Documentation	The presence of comprehensive and readable codebase documentation
Low-Level Manipulation	The justified use of inline assembly and low-level calls
Testing and Verification	The presence of robust testing procedures (e.g., unit tests, integration tests, and verification methods) and sufficient test coverage
Transaction Ordering	The system's resistance to transaction-ordering attacks

Rating Criteria	
Rating	**Description**
Strong	No issues were found, and the system exceeds industry standards.
Satisfactory	Minor issues were found, but the system is compliant with best practices.
Moderate	Some issues that may affect system safety were found.

Weak	Many issues that affect system safety were found.
Missing	A required component is missing, significantly affecting system safety.
Not Applicable	The category is not applicable to this review.
Not Considered	The category was not considered in this review.
Further Investigation Required	Further investigation is required to reach a meaningful conclusion.

B. Fix Review Results

When undertaking a fix review, Trail of Bits reviews the fixes implemented for issues identified in the original report. This work involves a review of specific areas of the source code and system configuration, not comprehensive analysis of the system.

On March 18, 2024, Trail of Bits reviewed the fixes and mitigations implemented by the Puffer team for the issues identified in this report. We reviewed each fix to determine its effectiveness in resolving the associated issue.

In summary, of the six issues described in this report, the Puffer team has resolved four issues, and has not resolved the remaining two issues. For additional information, please see the Detailed Fix Review Results below.

ID	Title	Status
1	An attacker can cause a DOS for withdrawals	Resolved
2	setWithdrawalLimit does not reset assetsWithdrawnToday	Resolved
3	stETH rebasing creates arbitrage opportunities	Unresolved
4	Vault withdrawals and redemptions can revert	Unresolved
5	ValidatorTicket purchase can revert due to strict equality	Resolved
6	updateDailyWithdrawals does not emit an event	Resolved

Detailed Fix Review Results

TOB-PUF-1: An attacker can cause a DOS for withdrawals
Resolved in PR #66. The contracts now use transient storage to ensure that a deposit and a withdrawal cannot occur within the same transaction.

TOB-PUF-2: setWithdrawalLimit does not reset assetsWithdrawnToday
Resolved in PR #67. The `assetsWithdrawnToday` state variable is now always correctly reset when a new day begins.

TOB-PUF-3: stETH rebasing creates arbitrage opportunities
Unresolved. The Puffer team provided the following context for this finding's fix status:

> *Acknowledged. We will not be using stETH for much longer and will be moving the assets to native ETH to be provisioning new validators.*

TOB-PUF-4: Vault withdrawals and redemptions can revert
Unresolved. The Puffer team provided the following context for this finding's fix status:

> *Acknowledged. This is an expected behavior and the withdrawal limits are implemented to prevent high volume withdrawals. There's also the option to withdraw using secondary markets.*

TOB-PUF-5: ValidatorTicket purchase can revert due to strict equality
Resolved in PR #190. The code will now not revert if a user does not send an exact multiple of the validator ticket price; in addition, the code will use the entire `msg.value` when computing the mint amount.

TOB-PUF-6: updateDailyWithdrawals does not emit an event
Resolved in PR #65. The `updateDailyWithdrawals` function now emits the `AssetsWithdrawnToday` event.

C. Fix Review Status Categories

The following table describes the statuses used to indicate whether an issue has been sufficiently addressed.

Fix Status	
Status	**Description**
Undetermined	The status of the issue was not determined during this engagement.
Unresolved	The issue persists and has not been resolved.
Partially Resolved	The issue persists but has been partially resolved.
Resolved	The issue has been sufficiently resolved.