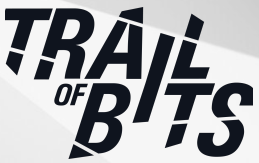

Storm Labs: Cove Boosties and coveYFI

Security Assessment (Summary Report)

March 19, 2024

Prepared for:
Sunil Srivatsa
Storm Labs

Prepared by: **Alexander Remie and Robert Schneider**

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:

Sam Greenup, Project Manager
sam.greenup@trailofbits.com

The following engineering director was associated with this project:

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

The following consultants were associated with this project:

Alexander Remie, Consultant
alexander.remie@trailofbits.com

Robert Schneider, Consultant
robert.schneider@trailofbits.com

Project Timeline

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

Date	Event
December 11, 2023	Pre-project kickoff call
December 18, 2023	Delivery of report draft; report readout meeting
January 25, 2024	Delivery of report with fix review appendix
March 19, 2024	Delivery of report with updated title

Executive Summary

Engagement Overview

Storm Labs engaged Trail of Bits to review the security of the Cove Boosties and coveYFI smart contracts. These smart contracts allow users to receive the benefit of YearnV3 (boosted dYFI yield) without requiring both veYFI and gauge tokens.

A team of two consultants conducted the review from December 11 to December 15, 2023, for a total of two engineer-weeks of effort. With full access to the source code and documentation, we performed static and dynamic testing of the repository, using automated and manual processes.

Observations and Impact

We managed to cover the entire codebase during the allotted time of this review. However, due to the large number of external contracts that are interacted with throughout the protocol, additional time could be spent to further dig into those interactions (i.e., how problems in any of these contracts could affect the Cove Boosties and coveYFI system). Also, the correct rounding of the various arithmetic operations in the protocol could use additional review.

Overall, the system is well designed and adequately documented. We identified several findings during this review. Two issues could result in a temporary DoS of the system due to missing or insufficient validation of values (in the Chainlink response and the reward duration update function). One other issue highlights the inadequate protection of rescuable funds from the `CoveYFI` contract, allowing funds of a specific token that should not be withdrawable to be withdrawn.

Recommendations

We recommend fixing the reported findings and taking the following additional actions:

- **Implement invariant testing of the system.** This will help to further ensure the correct functioning of the protocol, in terms of both system-wide invariants and the correctness of the arithmetic rounding.
- **Improve the documentation, as explained in the various Code Maturity Evaluation categories.**
- **Design an incident response plan for the coveYFI protocol.** This will improve the Storm Labs team's ability to deal with incidents. Additionally, plan regular dry runs of the various scenarios outlined in the incident response plan.

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 arithmetic operations in the system use Solidity version 0.8 or higher, which includes overflow protection. This improves the contract's resistance against common arithmetic vulnerabilities such as overflows and underflows. The code minimizes the use of unchecked blocks, reducing risks. However, there is one instance of an unchecked block that lacks an explanatory docstring. Including a docstring would improve the code's clarity and reliability. The testing suite covers functional correctness and reliability for arithmetic functions. However, there are gaps in the documentation, such as sufficient explanations for rounding directions and divide-by-zero operations. This lack of detailed documentation may cause confusion in understanding the contract's behavior in edge cases. While the arithmetic operations of the contract are solid, enhancing the documentation and providing explicit explanations, especially in complex or critical sections, would improve clarity and reliability.	**Moderate**
Auditing	The system emits events for critical operations, ensuring that significant state changes are traceable and observable. These events are not reused for different operations, which helps maintain event integrity and clarity. The team has documented plans for establishing an off-chain monitoring system to detect unexpected behavior. However, there is room for improvement in terms of monitoring documentation and alerting for users. It is crucial to provide users with detailed documentation explaining the purpose of events, their intended use, and their underlying assumptions. This will enable users to effectively monitor and troubleshoot problems on their own. Additionally, the Storm Labs	**Moderate**

	team does not have an incident response plan in place. Having such a plan is essential to outline procedures and responsibilities in the event of a contract anomaly or security incident. It ensures a swift and effective response to protect users and contract integrity. While the contract demonstrates good practices in event management and off-chain monitoring, improving the comprehensive documentation and implementing a robust incident response strategy will enhance the contract's overall security and the team's operational preparedness.	
Authentication / Access Controls	The system includes access controls for privileged functions, with specific roles assigned to privileged users. These roles are additionally protected by a multisig wallet, which requires multiple signatures to execute a transaction. Detailed documentation is provided for each role, outlining its associated privileges. Furthermore, the contract allows for the revocation of roles. The access controls have undergone testing to ensure their effectiveness.	**Satisfactory**
Complexity Management	Functions are designed with a clear scope and do not include excessive nesting. The code is free from redundancy or duplication. Inputs are well defined with necessary validations in place. Each function has a clear purpose and is accompanied by clear documentation. Core functions are tested. However, the naming conventions in the codebase lack sufficient documentation; adding such documentation will ensure that code elements are always named clearly and consistently.	**Satisfactory**
Cryptography and Key Management	The way that admin keys are managed, distributed, and protected was not in scope for this audit.	**Not Applicable**
Decentralization	The contract uses multisig wallets to prevent centralized control and to ensure that no single user can unilaterally impact the system. There are plans to transfer control of certain roles to a protocol governance mechanism. However, the absence of a time lock mechanism may not provide users with enough time to respond to system updates, including contract upgrades. In some instances,	**Weak**

	a single entity has control over user funds. While risks associated with trusted parties are well documented, documentation of risks related to third parties, such as Yearn, could be improved. The system's privileges are thoroughly documented.	
Documentation	The provided documentation is clear and thorough, including diagrams that describe the high-level system architecture and important user flows. NatSpec and inline comments are effectively used to aid in understanding the system's expected behavior. However, the documentation lacks details regarding known risks and system limitations. An incident response plan and explanations of how external incidents, such as those in Yearn, might impact the system need to be documented. Additionally, the documentation of function and system invariants needs enhancement to provide a more comprehensive understanding of the contract's operational boundaries and constraints.	**Moderate**
Low-Level Manipulation	No low-level manipulation is used in this codebase.	**Not Applicable**
Testing and Verification	The codebase has a comprehensive unit test suite that achieves nearly 100% coverage of branches and statements. Testing is integrated into a CI/CD pipeline. Automated testing is specifically applied to critical components. Integration tests are implemented to ensure seamless communication between the protocol and relevant third-party systems, and the testing suite validates functionality using real on-chain data. We recommend designing invariants for the system and testing them using Echidna. This will further ensure that the system works correctly when given various inputs.	**Moderate**
Transaction Ordering	Transaction ordering risks are limited and justified for user operations. The inclusion of maximal extractable value (MEV) is inherent to this type of protocol. Mitigations such as slippage protection have been implemented to prevent potential losses or manipulation. However, the system currently lacks specific tests for transaction ordering risks and could benefit from improved oracle data validation to safeguard against misreported data.	**Moderate**

Summary of Findings

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

ID	Title	Type	Severity
1	Missing validation in setRewardsDuration	Data Validation	**Medium**
2	YFI tokens can be moved out of CoveYFI using Rescuable	Access Controls	**High**
3	Missing validations of Chainlink response	Data Validation	**Medium**

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 January 19, 2024, Trail of Bits reviewed the fixes and mitigations implemented by the Storm Labs team for the issues identified in this report. We reviewed each fix to determine its effectiveness in resolving the associated issue.

In summary, Storm Labs has resolved all three issues disclosed in this report. For additional information, please see the Detailed Fix Review Results below.

ID	Title	Status
1	Missing validation in setRewardsDuration	**Resolved**
2	YFI tokens can be moved out of CoveYFI using rescuable	**Resolved**
3	Missing validations of Chainlink response	**Resolved**

Detailed Fix Review Results

TOB-COVE-1: Missing validation in setRewardsDuration
Resolved in PR #141. The `setRewardsDuration` function has been updated. Now, if the `rewardsDuration_` argument is zero, it will throw a `RewardDurationCannotBeZero` error. Similarly, if the `rewardsDuration` value for the provided `stakingToken` argument is zero, the function will throw a `StakingTokenNotAdded` error. Unit tests have been added to cover these new changes.

TOB-COVE-2: YFI tokens can be moved out of CoveYFI using rescuable
Resolved. The client provided the following context for this finding's fix status:

> *In our intended design and usage, `CoveYFI` should never have YFI tokens in them. Within the `CoveYFI.deposit()` call, the YFI tokens will be transferred from the user, to CoveYFI, to YSD, then to veYFI.*
>
> *Thus, the assumption that users' YFI deposits are under the risk via the rescue function is incorrect.*
>
> *All cases where YFI balance > 0 are when users do ERC-20 transfers to the `CoveYFI` contract.*
>
> *Therefore, we will keep the rescue function as is in `CoveYFI.sol`.*

TOB-COVE-3: Missing validations of Chainlink response
Resolved in PR #142. The `_getLatestPrice` function has been updated to include important validations on responses received from Chainlink oracles. If the `price` variable is returned with a value of zero, a `PriceFeedReturnedZeroPrice` error will be thrown. Additionally, if the `roundId` variable is greater than the `answeredInRound` variable, a `PriceFeedIncorrectRound` error will be thrown. Unit tests have been added to verify the expected behavior of these changes.

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.