Intuition Smart Contracts

Security Assessment (Summary Report)

April 8, 2024

Prepared for:
Billy Luedtke
Intuition

Prepared by: **Michael Colburn and Kurt Willis**

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

This report is considered by Trail of Bits to be business confidential information; it is licensed to Intuition under the terms of the project statement of work and intended solely for internal use by Intuition. Material within this report may not be reproduced or distributed in part or in whole without the express written permission of Trail of Bits.

The sole canonical source for Trail of Bits publications, if published, is the Trail of Bits Publications page. Reports accessed through any source other than that page may have been modified and should not be considered authentic.

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:

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 consultants were associated with this project:

Michael Colburn, Consultant
michael.colburn@trailofbits.com

Kurt Willis, Consultant
kurt.willis@trailofbits.com

Project Timeline

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

Date	Event
February 29, 2024	Technical onboarding call
March 7, 2024	Pre-project kickoff call
March 17, 2024	Delivery of report draft
March 17, 2024	Report readout meeting
March 28, 2024	Delivery of summary report
April 8, 2024	Delivery of summary report with fix review appendix

Project Targets

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

intuition-tob-audit

Repository	https://github.com/0xIntuition/intuition-tob-audit

Version	`8e80c751600c481164415f7a7f225217cc164ad2`
	`edc45845db0246c57b538ed5ab6e1f32becba89c`

Type	Solidity
Platform	EVM

Executive Summary

Engagement Overview

Intuition engaged Trail of Bits to review the security of its Intuition protocol's smart contracts. The codebase is composed of three core contracts: a standard contract, `TransparentUpgradeableProxy`; a wallet contract, `AtomWallet`, which leverages ERC-4337 account abstraction; and the main entry point and home of most protocol logic, the `EthMultiVault` contract, which takes inspiration from the ERC-4626 and ERC-1155 standards.

A team of two consultants conducted the review from March 10 to March 15, 2024, for a total of two engineer-weeks of effort. With full access to source code and documentation, we performed static and dynamic testing of the codebase, using automated and manual processes.

Observations and Impact

Most of our effort was focused on the `EthMultiVault` contract because it contains all the protocol-specific logic. We reviewed the bookkeeping, fee calculations, and flow of funds through the vault to identify any inconsistencies in the system's accounting or any known issues pertaining to ERC-4626 vaults. We also analyzed each of the contracts via static analysis with Slither and performed manual review to identify any standard Solidity issues or gaps in the access control mechanisms that could result in an attacker gaining elevated privileges.

Overall, the codebase is quite complex for its size, though by leveraging existing standards for the core building blocks of the contracts, this is largely isolated to the protocol-specific logic. We identified five medium-severity, six low-severity, and nine informational-severity issues as part of this review. Four of the medium-severity issues describe errors in the vault's bookkeeping that result in users receiving too few shares or portions of deposits becoming trapped in the vault. Several of the informational-severity issues were instances of redundant code or logic duplicated across multiple functions, which should be consolidated. Due to the number of issues in the bookkeeping and the amount of simplification that could be done, additional review may be beneficial.

Recommendations

Based on the codebase maturity evaluation and findings identified during the security review, Trail of Bits recommends that Intuition take the following steps

- Reduce the complexity of the codebase by removing duplicate code, redundant statements, and unnecessary features.

- After addressing the issues identified as part of this review, continue to improve the protocol’s test suite coverage to test for realistic scenarios, check for edge cases, explore more advanced testing techniques, and prevent regressions.
- Consider conducting an additional security review after carrying out the recommendations above.

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 contracts use only basic arithmetic operations but apply them in multiple layers, which results in fee calculation and bookkeeping logic that is at times difficult to follow. We identified five issues that would result in an incorrect number of shares being allocated or in assets being left unrecoverable in the `EthMultiVault` contract itself.	**Weak**
Auditing	The contracts emit appropriate events for any important state-modifying functions. However, compressed atom creation emits the compressed URI instead of the decoded value, which may be a more useful value.	**Satisfactory**
Authentication / Access Controls	The contracts have a straightforward access control schema, with one privileged role in each. We did not identify any issues related to access controls.	**Satisfactory**
Complexity Management	The wallet and proxy contracts are very straightforward. The vault contract contains most of the protocol's logic. The vault is broken down into logical functions, but there is a large amount of code duplication across functions that perform similar tasks. These should be consolidated into relevant internal helper functions. The fee logic is also fairly complex, with multiple different fees potentially being applied on different portions of a deposit at different times in a transaction. Additionally, referring to the atom shares purchased through calls to the `depositTriple` function as an `atomEquityFeeAmount` fee when this is not actually intended as a fee makes the flow of funds through the protocol more difficult to follow.	**Moderate**

Cryptography and Key Management	This category was not applicable for this review.	**Not Applicable**
Decentralization	The `EthMultiVault` contract is upgradeable. Almost all the parameters can be updated immediately after deployment by the contract admin. The protocol funds could be transferred out at any time through an upgrade if the admin's private key were compromised. The protocol's documentation indicates this admin will be set to an externally owned account (EOA) until sometime post-deployment ,when it will transition to a multisignature account. We strongly suggest migrating to a multisignature account, even one entirely controlled by core team members, during the production deployment process.	**Weak**
Documentation	The protocol documentation is thorough and includes definitions of protocol-specific terms as well as detailed descriptions of each of the main operations that can be executed through the vault. The contracts themselves also include NatSpec comments and helpful inline comments.	**Satisfactory**
Low-Level Manipulation	There is just one line of assembly in the `AtomWallet` contract that performs error handling after a low-level `call`. There are many other functions in the vault that use `call` to transfer ether between contracts or as part of the redemption process. We also observed one issue related with the use of the `abi.encodePacked` function and multiple variable-length arguments that would result in hash collisions that would allow an attacker to squat on triple identifiers.	**Satisfactory**
Testing and Verification	The codebase has decent test coverage and the beginnings of some exploration of invariant testing. However, several of the issues we identified could have been caught by a more comprehensive test suite.	**Moderate**
Transaction Ordering	We did not identify any issues related to transaction ordering. The vault contract uses ghost shares and internal bookkeeping to mitigate the known ERC-4626 share inflation attack.	**Satisfactory**

Summary of Findings

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

ID	Title	Severity
1	createAtomCompressed allows creating duplicate atoms with the same URI	Low
2	Upgrade could lead to mismatch in atom wallet address prediction	Low
3	Salt contains superfluous address(this)	Informational
4	Unbound storage reads in getVaultStates	Informational
5	EthMultiVault is missing ERC-4626 functionality	Informational
6	Protocol deposit fees are unaccounted for in createAtom	Medium
7	createAtom mints sharesForZeroAddress twice	Low
8	Redundant and ineffective reinitialization check	Informational
9	Impossible condition	Informational
10	Triple identifiers can contain hash collisions	Medium
11	EthMultiVault should not receive ether donations	Low
12	Atom equity should be calculated on raw asset amounts	Medium
13	Distributing atom equity should not include protocol fees	Medium
14	Distributing atom equity should not mint new shares to receiver	Informational
15	Atom wallets can be created before the atom is created	Low
16	Atom URI data is unbounded	Low

17	getVaultStates does not retrieve counter vaults	**Informational**
18	Excessive duplicate code	**Informational**
19	Admin can bypass fee setter limits	**Informational**
20	Asset accounting should not be reduced by minShare	**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. 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.

C. 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 April 2, 2024, Trail of Bits reviewed the fixes and mitigations implemented by the Intuition 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 20 issues described in this report, Intuition has resolved 19 issues and has not resolved the one remaining issue. For additional information, please see the Detailed Fix Review Results below.

ID	Title	Severity
1	createAtomCompressed allows creating duplicate atoms with the same URI	Resolved
2	Upgrade could lead to mismatch in atom wallet address prediction	Resolved
3	Salt contains superfluous address(this)	Resolved
4	Unbound storage reads in getVaultStates	Resolved
5	EthMultiVault is missing ERC-4626 functionality	Unresolved
6	Protocol deposit fees are unaccounted for in createAtom	Resolved
7	createAtom mints sharesForZeroAddress twice	Resolved
8	Redundant and ineffective reinitialization check	Resolved
9	Impossible condition	Resolved
10	Triple identifiers can contain hash collisions	Resolved
11	EthMultiVault should not receive ether donations	Resolved

12	Atom equity should be calculated on raw asset amounts	**Resolved**
13	Distributing atom equity should not include protocol fees	**Resolved**
14	Distributing atom equity should not mint new shares to receiver	**Resolved**
15	Atom wallets can be created before the atom is created	**Resolved**
16	Atom URI data is unbounded	**Resolved**
17	getVaultStates does not retrieve counter vaults	**Resolved**
18	Excessive duplicate code	**Resolved**
19	Admin can bypass fee setter limits	**Resolved**
20	Asset accounting should not be reduced by minShare	**Resolved**

Detailed Fix Review Results

TOB-INTUITION-1: createAtomCompressed allows creating duplicate atoms with the same URI
Resolved in commit 4d0b2ba. The standalone functions for operating on compressed atom URIs have been removed, and the functionality was replaced by using Solady's `LibZip.cdFallback` function in the contract's fallback function to handle decompressing the calldata.

TOB-INTUITION-2: Upgrade could lead to mismatch in atom wallet address prediction
Resolved in PR #38. Rather than deploying the atom vault contracts directly, the `EthMultiVault` contract now deploys a proxy contract that retrieves the wallet implementation contract address from a beacon, which will mitigate this issue as long as the proxy itself is not upgraded in the future.

TOB-INTUITION-3: Salt contains superfluous address(this)
Resolved in PR #27. The contract's address is no longer included in the calculation of the salt value.

TOB-INTUITION-4: Unbound storage reads in getVaultStates
Resolved in PR #25. The `getVaultStates` function was deemed unnecessary and removed from the codebase.

TOB-INTUITION-5: EthMultiVault is missing ERC-4626 functionality
Unresolved. The issue has not been resolved. Because the contracts are upgradeable, the Intuition team has indicated that this is intended behavior and that they may add in the missing functionality in future versions of the protocol.

The Intuition team provided the following context for this finding's fix status:

> *We appreciate the audit's detailed assessment and understand the concern regarding `EthMultiVault`'s alignment with the ERC-4626 standard, particularly the omission of certain functionalities like `previewMint` and `previewWithdraw`, and modifications in the return values of `previewRedeem`. Our approach to selectively implement the ERC-4626 standard functionalities was a deliberate choice, driven by the aim to cater efficiently to our users' immediate needs and to simplify the protocol's initial deployment. This decision took into account the potential complexities and associated risks that a full suite of functionalities could introduce, as well as the need to adhere to our development timeline.*
>
> *The `EthMultiVault` contract's design includes upgradeability as a core feature, allowing us to iteratively introduce additional functionalities, such as the mint and withdraw flows and their related helper functions, in future versions. This forward-looking approach positions us to enhance our protocol's features and its*

> *adherence to the ERC-4626 standard incrementally, without compromising on current operations or user experience.*

TOB-INTUITION-6: Protocol deposit fees are unaccounted for in createAtom
Resolved in commit `edc4584`. The protocol deposit fee is now subtracted properly from the deposit amount when creating atoms.

TOB-INTUITION-7: createAtom mints sharesForZeroAddress twice
Resolved in commit `edc4584`. Ghost shares for the zero address are now minted only once according to a Boolean flag that checks whether the recipient is an atom wallet.

TOB-INTUITION-8: Redundant and ineffective reinitialization check
Resolved in PR #28. The unnecessary reinitialization check has been removed.

TOB-INTUITION-9: Impossible condition
Resolved in PR #26. The check for an impossible condition (that an unsigned integer is not less than `0`) has been removed.

TOB-INTUITION-10: Triple identifiers can contain hash collisions
Resolved in PR #33. Triple identifiers are now determined by hashing the underlying atom IDs instead of the atom URIs.

TOB-INTUITION-11: EthMultiVault should not receive ether donations
Resolved in PR #24. The `receive` function was removed from the contract.

TOB-INTUITION-12: Atom equity should be calculated on raw asset amounts
Resolved in commit `028748d`. The atom equity amount is now calculated consistently between the `depositTriple` and `_depositIntoVault` functions.

TOB-INTUITION-13: Distributing atom equity should not include protocol fees
Resolved in commit `028748d`. The protocol fee is now accounted for when the `_distributeAtomEquity` function calls the `_depositIntoVault` function.

TOB-INTUITION-14: Distributing atom equity should not mint new shares to receiver
Resolved in PR #39. The name of the atom equity fee variable and its related internal function have been updated to clarify that it is not a fee in the traditional sense, but rather a portion of the deposit that goes towards shares of the triple's underlying atoms.

TOB-INTUITION-15: Atom wallets can be created before the atom is created
Resolved in PR #29. The `deployAtomWallet` function now performs a check to validate that the ID passed into it is nonzero but does not exceed the current value of the contract's `count` variable.

TOB-INTUITION-16: Atom URI data is unbounded
Resolved in PR #32. The EthMultiVault contract now takes a configurable atomUriMaxLength parameter that acts as an upper bound on the URI length supported by the protocol.

TOB-INTUITION-17: getVaultStates does not retrieve counter vaults
Resolved in PR #25. The getVaultStates function was deemed unnecessary and removed from the codebase.

TOB-INTUITION-18: Excessive duplicate code
Resolved in PR #30. The external entry points for the contract have been simplified. The compressed atom functions are handled implicitly via the fallback function's use of the LibZip.cdFallback function. Additionally, single and batch creation operations use a common internal function, and some deposit and redemption validation logic that was common to atoms and triples was moved into shared internal functions.

TOB-INTUITION-19: Admin can bypass fee setter limits
Resolved in PR #34. The setFeeDenominator function has been removed, preventing the contract's fee denominator from being set directly outside of a contract upgrade, and the various fee setter functions now perform their validation relative to the contract's feeDenominator value instead of hard-coded constants.

TOB-INTUITION-20: Asset accounting should not be reduced by minShare
Resolved in PR #36. The asset accounting in the _depositOnVaultCreation function now properly tracks the assets backing the minted shares for the zero address.

D. 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.