TRIBYTE

WAND PROTOCOL

SECURITY AUDIT REPORT

JULY 2025

Every Byte Builds Immutable Trust

No. 202507310001

Contents

Summary

Executive Summary

Tribyte conducted a comprehensive security audit of the Wand Protocol contracts throughout July 2025.Wand Protocol is a DeFi protocol that allows users to achieve yield enhancement, interest rate arbitrage, and other functionalities by purchasing yield rights with a small amount of capital. With the objective of identifying vulnerabilities and ensuring the robustness of the codebase. The audit encompassed both core contracts and their dependencies, delivering a thorough evaluation of the project's security posture.

The assessment leveraged a dual methodology, combining Manual Review and Static Analysis to meticulously examine the contracts. Our team adopted a multifaceted testing strategy, integrating black-box, gray-box, and white-box techniques to simulate real-world attack scenarios and detect potential weaknesses. Black-box testing evaluated the contracts from an external attacker's perspective, gray-box testing probed internal behaviors using specialized scripting tools, and white-box testing featured an in-depth, line-by-line code review by our experts.

Project Summary

Project Name	Wand Protocol
Language	Solidity
Github Link	https://github.com/wandfi/yield-vault-v2
Commit Hash	4ba0e94628c209ea74c476043dc5d2cb81a358ae
Deployment Address (BSC)	

Vulnerability Summary

Severity Level	Summary
Critical	0
High	0
Medium	1

Low	0
Informational	1

Findings

[CD-01] Centralization Risk in Protocol Owner

Category	Contract Design
Severity	Medium
Location	src/ProtocolOwner.sol
Status	Mitigated. The project states to transfer protocol ownership to a multi-signature wallet created via https://safe.global.

Description

The protocol concentrates critical control in a single owner (and closely related upgrader/operator roles controlled by the owner). This central point can:

1. Unilaterally upgrade UUPS implementations
2. Change critical parameters in ProtocolSettings
3. Pause/unpause core flows, update hooks/oracles/strategies, and move funds where allowed

This model introduces a single point of failure and governance risk.

Recommendation

We recommend migrating the owner address using tools like:

1. Multisig Wallet: Use 2/3 or 3/5 Gnosis Safe Wallet for owner/upgrade roles.
2. Timelock: Enforce a 24-72h delay for high-impact actions (upgrades, fee/threshold changes, treasury routes).

[CC-01] Inconsistent Function Naming Convention and Typo in Error Messages

Category	Coding Convention
Severity	Informational
Location	src/Protocol.sol:79 src/ProtocolSettings.sol:25

Description

The codebase contains inconsistent function naming conventions and a typo in error messages that affects code quality and maintainability.

1. Function Naming Inconsistency: The function updatedYieldContractFactory() uses past tense "updated" instead of the imperative "update", which is inconsistent with other similar functions in the same contract that use the imperative form (e.g., updateSettings(), updateYieldSwapHookHelper()).
2. Typo in Error Message: The error message "Zero address dectected" contains a typo where "dectected" should be "detected".

Recommendation

Rename the function to follow the established naming convention and fix the typo in the error message.

[CD-02] Large Function and Duplicate Code in unlockCallback

Category	Contract Design
Severity	Informational
Location	src/market/hooks/YieldSwapHook.sol:465-560

Description

The unlockCallback() function in YieldSwapHook.sol is excessively large (95+ lines) and contains significant code duplication. The function attempts to handle two different callback data types:

1. ForceTokenCallbackData - Custom callback data for force token operations
2. CallbackDataCustom - Standard BaseCustomCurve callback data.

The BaseCustomCurve implementation logic (lines 514-560) is directly copied from the parent contract instead of being properly inherited or extracted to a separate function. The function handles two completely different callback mechanisms, violating the Single Responsibility Principle.

Recommendation

Given that the implementation of unlockCallback in BaseCustomCurve is an external function and can't be called directly from the YieldSwapHook, we suggest:

1. Extract BaseCustomCurve Logic: Create an internal function _handleBaseCustomCurveCallback that contains the standard liquidity modification logic.
2. Separate Force Token Logic: Create an internal function _handleForceTokenCallback for the custom force token operations.
3. Refactor Main Function: Make unlockCallback a thin dispatcher that calls the appropriate internal function based on the data type

Appendix

Vulnerability Fix Status

Status	Description
Resolved	The project team has successfully implemented a complete fix to address the vulnerability, eliminating the associated risks. All identified issues have been rectified, and the codebase has been updated to ensure the vulnerability no longer poses a threat.
Mitigated	The project team has taken steps to reduce the impact or likelihood of the vulnerability being exploited, but the issue has not been completely resolved. While the risk has been partially addressed, some potential exposure may still remain, and further action is recommended.
Acknowledged	The project team has reviewed and confirmed the existence of the vulnerability but has chosen not to address or mitigate it at this time. This status indicates awareness of the issue, and the team may accept the associated risks or plan to address it in the future.
Declined	The project team has reviewed the reported vulnerability but determined it does not require action, either because they believe it poses no significant risk to the project or because it falls outside the project's scope or priorities. The issue remains unaddressed, and the associated risks are accepted by the team.

Vulnerability Severity Level

Level	Description
Critical	Critical severity vulnerabilities pose an immediate and severe threat to the project's security, potentially leading to significant loss of funds, unauthorized access, or complete system compromise. These issues must be addressed urgently before deployment or continued operation to ensure the safety of users and the integrity of the project.
High	High severity vulnerabilities can substantially impact the project's functionality, potentially enabling exploitation that results in loss of assets, data breaches, or disruption of critical operations. It is strongly recommended to prioritize and resolve these issues promptly to mitigate risks.
Medium	Medium severity vulnerabilities may affect the project's performance or security under specific conditions, potentially leading to inefficiencies, minor exploits, or degraded user experience. It is advisable to address these issues to enhance the overall robustness of the system.

Low	Low severity vulnerabilities have a minimal impact on the project's security or functionality and are unlikely to be exploited in typical scenarios. However, they may still pose theoretical risks. The project team should evaluate these issues and consider fixing them to improve long-term stability.
Informational	Informational findings do not directly impact the security or functionality of the project but highlight areas for improvement, such as adherence to best practices, code optimization, or architectural enhancements. Addressing these suggestions can lead to better maintainability and alignment with industry standards.

Audit Items

Categories	Audit Items
Coding Convention	Obsolete Code
	Debug Code
	Comments / Dev Notes
	Compiler Versions
	License Identifier
	Require / Revert / Assert Usage
	Contract Size
	Gas Consumption
	Event Emission
	Parameter Check
General Vulnerability	Centralization
	Denial of Service
	Reply Attack
	Reentrancy Attack
	Race Conditions
	Integer Overflow / Underflow
	Arithmetic Accuracy Deviation
	Array Index Out of Bounds

	Receive / Fallback Function
	Payable and msg.value Usage
	tx.origin Authentication
	ERC20 Token Decimals
	ERC20 Safe Transfer
	ERC721 Safe Transfer
	Rebasable Token Support
	Native Token Support
	Storage / Memory Usage
	Function Permissions
External Dependency	Oracle Usage
	External Protocol Interaction
Protocol Design	Economics Design
	Formula Derivation
Contract Design	Factory Contract
	Proxy Usage
	EIP2535 Diamond Pattern Usage
	Upgradability / Pluggability

Disclaimer

Tribyte issues this audit report based solely on the code and materials provided by the client up to the report's issuance date. We assume the provided information is complete, accurate, and untampered. Tribyte is not liable for any losses or issues arising from incomplete, altered, or concealed information, or from changes made after the audit.

This report evaluates only the specified smart contracts or systems within the agreed scope, using Tribyte's tools and methodologies. It does not endorse the project's business model, team, or legal status, nor does it guarantee the absence of vulnerabilities due to technical limitations. The report is for the client's use only and may not be shared, quoted, or relied upon by third parties without Tribyte's written consent.

Conclusion

During this security audit, we conducted a comprehensive examination of the Wand Protocol's smart contract design and implementation. Our assessment encompassed multiple security dimensions: we thoroughly reviewed all external dependencies (including mathematical libraries, role management systems, and other common vulnerability vectors) and identified no security concerns in these foundational components. We then analyzed the protocol's design documentation to gain a thorough understanding of the yield vault mechanism and its underlying architecture. Subsequently, we evaluated coding practices, business logic implementation, and overall security posture. Based on our comprehensive review, no major security vulnerabilities were identified, and the current deployment demonstrates adherence to industry best practices and security standards. As outlined in the Disclaimer section, we welcome constructive feedback and suggestions regarding our findings, audit procedures, and scope of examination.