QuillAudits

AUDIT REPORT

May , 2025

For

CYBRO

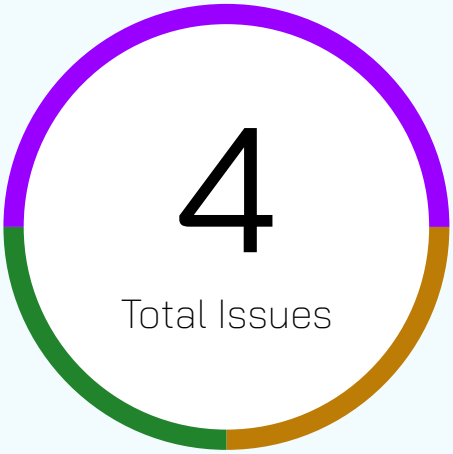
Table of Content

Executive Summary

Project name	Cybro
Project URL	https://cybro.io/
Overview	CYBRO is a multichain AI-powered yield aggregator offering users diversified investment options through a unified interface. CYBRO aggregates vaults of other protocols and its complex strategies wrapped in simple and easy-to-use One-Clicks. Under a user-centric philosophy, CYBRO prioritizes seamless processes, particularly in crucial areas such as portfolio management, onboarding, deposits, and withdrawals. Across is an investment fund based on BaseVault. Similarly Steer and Jones are DEX protocols on top of which cybro is building liquidity management vaults.
Audit Scope	The scope of this Audit was to analyze the Cybro Vault Smart Contracts for quality, security, and correctness.
Source Code link	https://github.com/cybro-io/vault-contracts/commit/fc0935c11bc90291828fc26cc6ed28ed4c85cf6a
Contracts in Scope	src/vaults/AcrossVault.sol src/vaults/SteerCamelotVault.sol src/vaults/JonesCamelotVault.sol
Branch	Main
Commit Hash	c0935c11bc90291828fc26cc6ed28ed4c85cf6a
Language	Solidity

Blockchain	EVM
Method	Manual Analysis, Functional Testing, Automated Testing
Review 1	5th May 2025 - 15th may 2025
Updated Code Received	20th May 2025
Review 2	20th May 2025 - 26th May 2025
Fixed In	cb4af160697210354601b392877050d01b812417

Number of Issues per Severity



High	0 (0.00%)
Medium	1 (25.00%)
Low	1 (25.00%)
Informational	2 (50.00%)

Severity

Issues	High	Medium	Low	Informational
Open	0	0	0	0
Resolved	0	**1**	**1**	**2**
Acknowledged	0	0	0	0
Partially Resolved	0	0	0	0

Checked Vulnerabilities

- ☑ Access Management
- ☑ Arbitrary write to storage
- ☑ Centralization of control
- ☑ Ether theft
- ☑ Improper or missing events
- ☑ Logical issues and flaws
- ☑ Arithmetic Computations Correctness
- ☑ Race conditions/front running
- ☑ SWC Registry
- ☑ Re-entrancy
- ☑ Timestamp Dependence
- ☑ Gas Limit and Loops
- ☑ Exception Disorder
- ☑ Gasless Send
- ☑ Use of tx.origin
- ☑ Malicious libraries
- ☑ Compiler version not fixed
- ☑ Address hardcoded
- ☑ Divide before multiply
- ☑ Integer overflow/underflow
- ☑ ERC's conformance
- ☑ Dangerous strict equalities
- ☑ Tautology or contradiction
- ☑ Return values of low-level calls
- ☑ Missing Zero Address Validation
- ☑ Private modifier
- ☑ Revert/require functions
- ☑ Multiple Sends
- ☑ Using suicide
- ☑ Using delegatecall
- ☑ Upgradeable safety
- ☑ Using throw

- [x] Using inline assembly
- [x] Unsafe type inference
- [x] Style guide violation
- [x] Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools And Platforms Used For Audit

Remix IDE, Foundry, Solhint, Mythril, Slither, Solidity statistical analysis.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational Issues

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

Security vulnerabilities identified that must be resolved and are currently unresolved.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/ impact of the security issue, but are not completely resolved.

Medium Severity Issues

Missing whenNotPaused modifier in redeem function

Resolved

Path

BaseVault.sol

Function

redeem()

Description

The BaseVault contract, implements a pause mechanism through the PausableUpgradeable contract to halt operations during emergencies. However, while the deposit() function includes the whenNot-Paused modifier, the redeem() function does not have this protection.

This creates inconsistent behaviour in the contract's emergency response capabilities. During a paused state, users cannot deposit funds (which is expected), but they can still redeem their shares.

```
function redeem(uint256 shares, address receiver, address owner, uint256 minAssets)
    public
    virtual //@audit whenNotPaused is not there
    returns (uint256 assets)
{
    if (_msgSender() != owner) {
        _spendAllowance(owner, _msgSender(), shares);
    }
    assets = _redeemBaseVault(shares, receiver, owner);
    require(assets >= minAssets, MinAssets());
}

```

Recommendation

Implement whenNotPaused in redeem also.

Low Severity Issues

Usage of answerInRound is deprecated

Resolved

Path

AcrossVault.sol

Function

_getPrice()

Description

Usage of answerInRound has been deprecated. Previously it was used for when answers could take multiple rounds to be computed.

Recommendation

To resolve the issue related to stale price check with updatedAt variable can be used.

Informational Severity Issues

Approve for each call instead of giving unlimited approval during initialization

Resolved

Path

JonesCamelotVault.sol, SteerCamelotVault.sol

Function

initialize()

Description

While this approach saves gas by eliminating the need for repeated approvals during deposit and withdrawal operations, it significantly increases the security risk profile of the contract. If either of the external integrations are compromised, attackers would have unlimited access to the vault's token0 and token1 holdings.

Recommendation

Implement a capped approval strategy with reasonable limits that get refreshed when needed. Replace unlimited approvals with exact-amount approvals before each operation

Redundant functionality in AcrossVault.sol

Resolved

Path

src/vaults/AcrossVaul.sol

Function

_validateTokenToRecover()

Description

In contract AcrossVault, the _validateTokenToRecover() function is present which just returns true boolean value. As such there is no need for such a function and boolean value can directly be called instead of through a function. It'll also save gas.

Recommendation

To resolve the issue the function can be remove by making necessary changes to other function where it is called.

Functional Tests

Some of the tests performed are mentioned below:

- Is deposit amount correct
- Is reinvest is working correctly
- Is redeem function working correctly

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of Cybro V4. We performed our audit according to the procedure described above.

Issues of Medium, Low, Informational severities were found. Cybro team resolved them all.

Disclaimer

At QuillAudits, we have spent years helping projects strengthen their smart contract security. However, security is not a one-time event—threats evolve, and so do attack vectors. Our audit provides a security assessment based on the best industry practices at the time of review, identifying known vulnerabilities in the received smart contract source code.

This report does not serve as a security guarantee, investment advice, or an endorsement of any platform. It reflects our findings based on the provided code at the time of analysis and may no longer be relevant after any modifications. The presence of an audit does not imply that the contract is free of vulnerabilities or fully secure.

While we have conducted a thorough review, security is an ongoing process. We strongly recommend multiple independent audits, continuous monitoring, and a public bug bounty program to enhance resilience against emerging threats.

Stay proactive. Stay secure.

About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem

7+ Years of Expertise	**1M+** Lines of Code Audited
$30B+ Secured in Digital Assets	**1400+** Projects Secured

Follow Our Journey

AUDIT REPORT

May , 2025

For

CYBRO

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com audits@quillaudits.com