QuillAudits

AUDIT REPORT

December, 2024

For

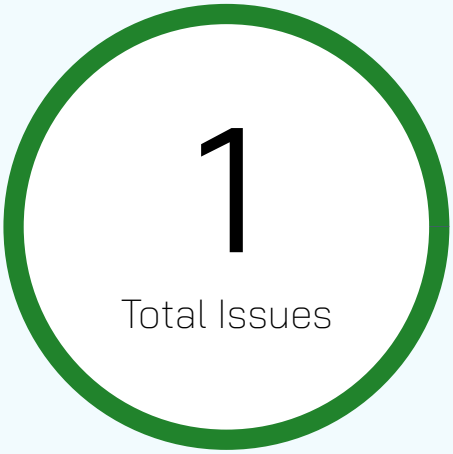
Table of Content

Executive Summary

Project name	Iplaza Labs
Project Overview	The Plaza (ISPZ) project implements a sophisticated token trading system using a bonding curve mechanism. The core BondingCurve contract allows users to buy and sell PAZA tokens using USDC, with prices dynamically adjusted based on the token supply. It incorporates a custom BondingCurve-Calculations library for complex mathematical operations, and includes features like tax accumulation and claiming.
Audit Scope	The Scope of the Audit is to analyse IPlaza contracts changes for security, code quality and correctness.
Contracts under Scope	https://github.com/rahul-ray30/PazaBondingCurve/commit/cfcf79f4122f604df35078e7f8971e731a3bf472 https://github.com/rahul-ray30/Polygon-Bridge-Contracts/commit/d16b669fa1b0009893ae50d1c293baf64c6cf6cb
Language	Solidity
Blockchain	Polygon
Method	Manual Review, Functional Testing, Automated Testing, etc. All the raised flags were manually reviewed and re-tested to identify any false positives.
First Review	16th December 2024 - 17th December 2024
Updated Code Received	18th December 2024

Second Review	18th December 2024
Fixed In	https://github.com/rahul-ray30/PazaBondingCurve/commit/1e6d896a439e76fc781121da49c4a61efa4553dc
Note	The Scope of the Audit is limited to changes made to Bonding curve and Polygon Bridge Contract. The Changes are as Follows: 1)Gelato Implementation meta transaction instead of Biconomy in Bonding Curve Contract 2) Polygon Bridge Contract is made upgradable.

Number of Issues per Severity



High	0 (0.00%)
Medium	0 (0.00%)
Low	1 (100.00%)
Informational	0 (0.00%)

Severity

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

Checked Vulnerabilities

- ☑ Re-entrancy
- ☑ Timestamp Dependence
- ☑ Gas Limit and Loops
- ☑ Transaction-Ordering Dependence
- ☑ DoS with Block Gas Limit
- ☑ Use of tx.origin
- ☑ Exception Disorder
- ☑ Balance Equality
- ☑ Gasless Send
- ☑ Byte Array
- ☑ Transfer Forwards All Gas
- ☑ Compiler Version Not Fixed
- ☑ Redundant Fallback Function
- ☑ Send Instead of Transfer
- ☑ Style Guide Violation
- ☑ Unchecked External Call
- ☑ Unchecked Math
- ☑ Unsafe Type Inference
- ☑ Implicit Visibility Level
- ☑ Access Management
- ☑ Arbitrary Write to Storage
- ☑ Centralization of Control
- ☑ Improper or Missing Events
- ☑ Logical Issues and Flaws
- ☑ Arithmetic Computations Correctness
- ☑ Race Conditions/Front Running
- ☑ SWC Registry
- ☑ Ether Theft
- ☑ Malicious Libraries
- ☑ Address Hardcoded
- ☑ Divide Before Multiply
- ☑ Integer Overflow/Underflow

- [x] Dangerous Strict Equalities
- [x] Tautology or Contradiction
- [x] Return Values of Low-Level Calls
- [x] Missing Zero Address Validation
- [x] Private Modifier
- [x] Revert/Require Functions
- [x] Multiple Sends
- [x] Using Delegatecall
- [x] Upgradeable Safety
- [x] Using Throw
- [x] Using Inline Assembly

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 Issues

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

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.

Issue Status

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.

Low Severity Issues

Missing Check for Calldata length

Resolved

Path

contracts/BondingCurve.sol

Description

Best to have an check that the call data length is >= contextSuffixLength along with the isTrusted-Forwarder check in both _msgSender() and _msgData() functions. Also should cache the data length in a local variable and then use it to compare and perfrom operations.

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 the ISPZ Changes Review in contracts. We performed our audit according to the procedure described above. One issues of Low severity was found, which the IPlaza Labs Team Fixed it.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in IPlaza Labs smart contract. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of IPlaza Labs smart contract. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of the IPlaza Labs team to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

About QuillAudits

QuillAudits is a leading name in Web3 security, offering top-notch solutions to safeguard projects across DeFi, GameFi, NFT gaming, and all blockchain layers. With six years of expertise, we've secured over 1000 projects globally, averting over $30 billion in losses. Our specialists rigorously audit smart contracts and ensure DApp safety on major platforms like Ethereum, BSC, Arbitrum, Algorand, Tron, Polygon, Polkadot, Fantom, NEAR, Solana, and others, guaranteeing your project's security with cutting-edge practices.

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

Follow Our Journey

AUDIT REPORT

December, 2024

For

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com audits@quillaudits.com