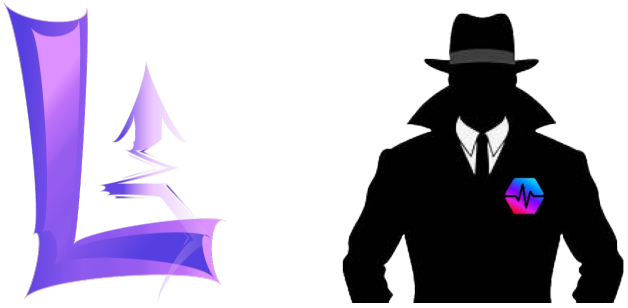
Pulse In Private - Security Review v2



03.04.2025

Conducted by:

Kann, Lead Security Researcher

Ivan Fitro, Lead Security Researcher

Table of Contents

1 About Kann

Kann a Security Reseacher and Founder of Kann Audits.

2 About Ivan Fitro

Ivan Fitro a Security Reseacher and Founding SR in Kann Audits.

3 Disclaimer

Audits are a time, resource, and expertise bound effort where trained experts evaluate smart contracts using a combination of automated and manual techniques to identify as many vulnerabilities as possible. Audits can show the presence of vulnerabilities **but not their absence**.

4 Risk classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4.1 Impact

- **High** - leads to a significant loss of assets in the protocol or significantly harms a group of users.
- **Medium** - only a small amount of funds can be lost or a functionality of the protocol is affected.
- **Low** - any kind of unexpected behaviour that's not so critical.

4.2 Likelihood

- **High** - direct attack vector; the cost is relatively low to the amount of funds that can be lost.
- **Medium** - only conditionally incentivized attack vector, but still relatively likely.
- **Low** - too many or too unlikely assumptions; provides little or no incentive.

4.3 Actions required by severity level

- **Critical** - client **must** fix the issue.
- **High** - client **must** fix the issue.
- **Medium** - client **should** fix the issue.
- **Low** - client **could** fix the issue.

5 Executive summary

Overview

Project Name	Pulse In Private
Repository	https://github.com/alexbabits/pip
Commit hash	b7234af
Resolution	Fixed
Documentation	https://pip-1.gitbook.io/pip-docs
Methods	Manual review

Scope

/src/Pip.sol
/circuits/height12/withdraw.circom

Issues Found

Critical risk	0
High risk	0
Medium risk	0
Low risk	0
Informational	2

6 Findings

6.1 Informational

6.1.1 Add Off-Chain Checks for requestWithdraw to Prevent Spam with Invalid Proofs and Potential Gas Waste for Relayers

Severity: *Informational*

Description: In the requestWithdraw process, users submit a withdrawal request by providing a proof and public signals which get sent in a Telegram group. Relayers monitor this group and use the provided information to call the on-chain withdraw function.

However, without proper off-chain validation, bad actors can flood the Telegram group with invalid or duplicate requests, causing relayers to waste gas attempting failed transactions.

Recommendation: To prevent this, off-chain validation should include:

Nullifier Existence – Ensuring the provided nullifier corresponds to a valid deposit.

Correct Recipient – Verifying that the recipient address matches the expected one for the nullifier.

Withdrawal Status – Checking if the nullifier has already been used for a withdrawal.

Resolution: Fixed

6.1.2 Add an On-Chain View Function for Relayers to Estimate Exact Fee Rewards

Severity: *Informational*

Description: Currently, relayers calling the withdraw function lack an efficient way to determine the exact fee they will receive for processing a withdrawal. This uncertainty may discourage participation or lead to inefficient relaying strategies.

Recommendation: To address this, an on-chain view function should be implemented to allow relayers to see the exact fee amount they will receive before executing a withdrawal transaction.

Resolution: Fixed