# Neutrl Security Review

## Pashov Audit Group

Conducted by: merlinboii, Udsen, Ch_301

March 27th 2025 - March 28th 2025

# Contents

# 1. About Pashov Audit Group

Pashov Audit Group consists of multiple teams of some of the best smart contract security researchers in the space. Having a combined reported security vulnerabilities count of over 1000, the group strives to create the absolute very best audit journey possible - although 100% security can never be guaranteed, we do guarantee the best efforts of our experienced researchers for your blockchain protocol. Check our previous work here or reach out on Twitter @pashovkrum.

# 2. Disclaimer

A smart contract security review can never verify the complete absence of vulnerabilities. This is a time, resource and expertise bound effort where we try to find as many vulnerabilities as possible. We can not guarantee 100% security after the review or even if the review will find any problems with your smart contracts. Subsequent security reviews, bug bounty programs and on-chain monitoring are strongly recommended.

# 3. Introduction

A time-boxed security review of the **Neutrl-lab/plp-contract** repository was done by **Pashov Audit Group**, with a focus on the security aspects of the application's smart contracts implementation.

# 4. About Neutrl

Neutrl implements SimpleDepositor smart contract to process ERC20 token deposits through customizable whitelists, automatically routing all assets to a designated custodian while maintaining deposit records. The system has flexible administration with asset/user permissions and emergency pause features.

# 5. Risk Classification

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

## 5.1. Impact

- High - leads to a significant material loss of assets in the protocol or significantly harms a group of users.
- Medium - only a small amount of funds can be lost (such as leakage of value) or a core functionality of the protocol is affected.
- Low - can lead to any kind of unexpected behavior with some of the protocol's functionalities that's not so critical.

## 5.2. Likelihood

- High - attack path is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount of funds that can be stolen or lost.
- Medium - only a conditionally incentivized attack vector, but still relatively likely.
- Low - has too many or too unlikely assumptions or requires a significant stake by the attacker with little or no incentive.

## 5.3. Action required for severity levels

- Critical - Must fix as soon as possible (if already deployed)
- High - Must fix (before deployment if not already deployed)
- Medium - Should fix
- Low - Could fix

# 6. Security Assessment Summary

*review commit hash* - c4d00d995e35d6e7ff7390290114022af01cf5f9

*fixes review commit hash* - 5877c828754e0371d6b7b7cfd6205409ac4777dd

## Scope

The following smart contracts were in scope of the audit:

- `SimpleDepositor`

# 7. Executive Summary

Over the course of the security review, merlinboii, Udsen, Ch_301 engaged with Neutrl to review Neutrl. In this period of time a total of **1** issues were uncovered.

## Protocol Summary

| | |
|---|---|
| **Protocol Name** | Neutrl |
| **Repository** | https://github.com/Neutrl-lab/plp-contract |
| **Date** | March 27th 2025 - March 28th 2025 |
| **Protocol Type** | Custodian Service |

## Findings Count

| **Severity** | **Amount** |
|---|---|
| Low | 1 |
| **Total Findings** | **1** |

## Summary of Findings

| **ID** | **Title** | **Severity** | **Status** |
|---|---|---|---|
| [L-01] | renounceOwnership can be called to break operations | Low | Resolved |

# 8. Findings

## 8.1. Low Findings

### [L-01] `renounceOwnership` can be called to break operations

All the state changing functions except `deposit` can only be called by the `Owner` of the `SimpleDepositor` contract. Hence the contract is highly centralized. Since `SimpleDepositor` is inheriting from `Ownable` it implements the `renounceOwnership()` function as shown below:

```
function renounceOwnership() public virtual onlyOwner {
        _transferOwnership(address(0));
    }
```

If this is called by the Owner maliciously or by mistake, it will completely break the `SimpleDepositor` contract since there is no Owner to manage the contract configurations and locked asset withdrawal.

Hence it is recommended to override the `renounceOwnership` to revert if called.