Mass Security Review

Pashov Audit Group

Conducted by: rvierdiiev, carrotsmuggler, Stefan Latinović

March 18th 2024 - March 24th 2024

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 **mass-core-tetris** repository was done by **Pashov Audit Group**, with a focus on the security aspects of the application's smart contracts implementation.

4. About Mass

Mass Smart Accounts are smart contract wallets created on the account abstraction protocol - Tetris. Account abstraction is achieved in the smart contract layer, through the use of the HyVM, an Ethereum Virtual Machine (EVM) Hypervisor. This enables the execution of arbitrary EVM bytecode without the need for deployed logic.

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 - ccaafae41483e8cc07ba253e542430705390dae7

fixes review commit hash - e9edf9d37e4ac82648f37c92005e2825340620ef

Scope

The following smart contracts were in scope of the audit:

- `ReentrancyGuardUpgradeable`
- `TimelockControllerEmergency`
- `SmartAccountFactoryUtils`
- `Errors`
- `FlashloanerMSA`
- `ImplementationResolver`
- `MassSmartAccount`
- `Proxy`
- `SmartAccountFactory`
- `SmartAccountOwnerResolver`
- `WithdrawerWeth`

7. Executive Summary

Over the course of the security review, rvierdiiev, carrotsmuggler, Stefan Latinović engaged with Mass to review Mass. In this period of time a total of **2** issues were uncovered.

Protocol Summary

Protocol Name	Mass
Repository	https://github.com/MassDotMoney/mass-core-tetris
Date	March 18th 2024 - March 24th 2024
Protocol Type	Account Abstraction

Findings Count

Severity	Amount
Low	2
Total Findings	**2**

Summary of Findings

ID	Title	Severity	Status
[L-01]	Immutable Aave pool address	Low	Resolved
[L-02]	Missing msg.value check in some functions	Low	Acknowledged

8. Findings

8.1. Low Findings

[L-01] Immutable Aave pool address

According to the Aave docs, the address of the Aave pool should be fetched from the pool address provider contract. The address of the pool address provider is never expected to be changed and is thus immutable. The suggestions can be found here in the docs.

Instead of saving the pool address in an immutable variable in the FlashloanerMSA.sol contract, consider instead saving the pool address provider contract address and calling `getAddress()` function to get the address of the pool

[L-02] Missing `msg.value` check in some functions

Some functions in the `MassSmartAccount.sol` contract are missing the check that `msg.value == value`. This check is required in functions where users can send eth when calling the function and makes sure that the user doesn't end up sending extra eth than they specified, leading to them leaving that eth in the account.

This issue is present in the functions `executeCall`, `executeHyVMCall` and `performReentrantCall`. The other functions handling eth have this check but not these few. Consider adding this check to those functions.