Ambire Security Review

Pashov Audit Group

Conducted by: pashov

October 20th, 2023

Contents

1. About pashov

Krum Pashov, or **pashov**, is an independent smart contract security researcher. Having found numerous security vulnerabilities in various protocols, he does his best to contribute to the blockchain ecosystem and its protocols by putting time and effort into security research & reviews. Check his 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 I try to find as many vulnerabilities as possible. I 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 **Pashov** protocol was done by **pashov**, with a focus on the security aspects of the application's smart contracts implementation.

4. About Ambire

Copied from the previous security reviews

Ambire is a smart wallet protocol. Users have wallets (accounts) which are controlled by them or other addresses that have "privileges" to do so. A user can do an off-chain signature of a bundle of transactions and anyone can execute it on-chain. Different signature schemes are allowed, for example EIP712, Schnorr, Multisig and others. The protocol works in a counterfactual manner, meaning a user wallet gets deployed only on its first transaction. The actual deployment is an EIP1167 minimal proxy for the wallet smart contract.

The `Ambire` protocol extended its signature validator options by adding an "external signature validator" option. One such option is the `DKIMRecoverySigValidator`, which is basically a way to recover access to your smart wallet by using your email. In the case that you have access & control over your secondary key and your email but you lost your primary key, you can instantly recover access to your account. If you have lost access/control over either of them you can still queue a recovery but you'd have to wait for a timelock to pass.

Continued

Ambire added ERC4337 support in their `AmbireAccount` contract with the `validateUserOp` functionality. Its implementation has a special caveat allowing an account to easily enable 4337 on it. There is also the new `AmbirePaymaster` contract which will allow users to delegate the gas costs for their transactions to it.

ERC4337 standard

Observations

The protocol has a special ERC4337 implementation that allows the `AmbireAccount` contract to not use the `UserOperation` signature field when it is about to call `executeMultiple` on the account.

The `AmbirePaymaster` contract omits functionality for staking to the ERC4337 system since it is not reading/writing from/to storage and doesn't have a `postOp` implementation.

Privileged Roles & Actors

- Paymaster - makes possible to cover user transaction gas costs

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* - da3ba641a004d1f0143a20ddde48049b619431ad**

***fixes review commit hash* - 62ba7dc8eaca4c1a1f66a777aecc475735449ef3**

7. Executive Summary

Over the course of the security review, pashov engaged with Ambire to review Ambire. In this period of time a total of **2** issues were uncovered.

Protocol Summary

Protocol Name	Ambire
Date	October 20th, 2023

Findings Count

Severity	**Amount**
Low	2
Total Findings	**2**

Summary of Findings

ID	**Title**	**Severity**	**Status**
[L-01]	The DKIM logic to verify headers allows weird cases	Low	Resolved
[L-02]	No withdrawTo functionality in AmbirePaymaster	Low	Resolved

8. Findings

8.1. Low Findings

[L-01] The DKIM logic to verify headers allows weird cases

The `_verifyHeaders` method in `DKIMRecoverySigValidator` now allows for the following two anomalies:

1. A valid set of headers that have extra text in between them, which is in between two `\r\n` expressions
2. Reordered `subject`, `to` and `from` headers are now allowed - previously the order - `from`, `to`, `subject` was expected

You can change the code to be a sequential state machine, basically enforcing an order of text in headers.

[L-02] No `withdrawTo` functionality in `AmbirePaymaster`

The ERC4337 implementation on Ethereum has a `StakeMaster` contract with a `withdrawTo` functionality, allowing a paymaster to withdraw his deposit as seen here. The issue is that `AmbirePaymaster` doesn't implement a direct way to call this functionality but it does, however, have the arbitrary call functionality allowed for the `relayer` address. Through that functionality the `withdrawTo` method can be called, but the `call` method has the following comment in its NatSpec:

```
* @notice  This method can be used to withdraw stuck tokens or airdrops
```

which means it wasn't expected to do so. If you plan on using `call` for other things as well, consider making it `payable` since it uses a `value` argument but

the contract doesn't have a way to receive ETH.