

Ambire, EIP-7702 Update Security Audit

Report Version 0.1

February 20, 2025

Conducted by **Hunter Security**

Table of Contents

1 About Hunter Security 3

2 Disclaimer 3

3 Risk classification 3

3.1 Impact 3

3.2 Likelihood 3

3.3 Actions required by severity level 3

4 Executive summary 4

5 Findings 5

5.1 Informational 5

5.1.1 Storage layout adjustment 5

1 About Hunter Security

Hunter Security is an industry-leading smart contract security company. Having conducted over 100 security audits protecting over \$1B of TVL, our team delivers top-notch security services to the best DeFi protocols. For security audit inquiries, you can reach out on Telegram or Twitter at [@georgehntr](#).

2 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 reveal the presence of vulnerabilities, but cannot guarantee their absence.

3 Risk classification

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

3.1 Impact

- **High** - leads to a significant loss of assets in the protocol or significantly harms a group of users.
- **Medium** - involves a small loss of funds or affects a core functionality of the protocol.
- **Low** - encompasses any unexpected behavior that is non-critical.

3.2 Likelihood

- **High** - a direct attack vector; the cost is relatively low compared to the potential loss of funds.
- **Medium** - only a conditionally incentivized attack vector, with a moderate likelihood.
- **Low** - involves too many or unlikely assumptions; offers little to no incentive.

3.3 Actions required by severity level

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

4 Executive summary

Overview

Project Name	Ambire, EIP-7702 Update
Repository	https://github.com/AmbireTech/ambire-common
Commit hash	de88e26041db8777468f384e56d5ad0cb96e29a5
Resolution	-
Methods	Manual review & testing

Scope

contracts/AmbireAccount.sol
contracts/AmbireAccount7702.sol
contracts/libs/Eip712HashBuilder.sol
contracts/deployless/IAmbireAccount.sol

Issues Found

High risk	0
Medium risk	0
Low risk	0
Informational	1

5 Findings

5.1 Informational

5.1.1 Storage layout adjustment

Severity: Informational

Files: AmbireAccount.sol, IAmbireAccount.sol

Description: The initialization bytecode for an AmbireAccount contract is currently crafted off-chain by attaching the privileged addresses and corresponding roles/levels to the contract's deployment bytecode. This is done by manually storing the according values in the contract's storage via SSTORE operations at the dedicated slots.

In this update, the privilege mapping has been moved from slot 0 to the following:

```
bytes32 constant AMBIRE_STORAGE_POSITION = keccak256("ambire.smart.contracts.storage");
```

Recommendation: Verify whether the adjustment in the storage slots for the *privileges* mapping is handled when crafting the initialization bytecode in the off-chain scripts.

Resolution: Acknowledged.