SECURITY ANALYSIS

by Pessimistic

This report is public

January 28, 2025

ABSTRACT

In this report, we consider the security of smart contracts of Agent A.I. project. Our task is to find and describe security issues in the smart contracts of the platform.

DISCLAIMER

The audit does not give any warranties on the security of the code. A single audit cannot be considered enough. We always recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. Besides, a security audit is not investment advice.

SUMMARY

In this report, we considered the security of Agent A.I. smart contracts. We described the audit process in the section below.

The audit showed several issues of medium severity:
The amount of sold tokens is not checked, Missing price age check,
Not all contracts are initialized, Owner role. Also, several low-severity issues were found.

The overall quality of the code is good.

GENERAL RECOMMENDATIONS

We recommend fixing the mentioned issues. In addition, consider either adding documentation to the repository or making it public.

PROJECT OVERVIEW

Project description

For the audit, we were provided with Agent A.I. project on a private GitHub repository, commit 41e8b59a2201a7754bbf770272a64c08d324362b.

The scope of the audit included the whole repository.

The documentation was provided in the chat with the developers.

All 37 tests pass successfully after the fixes. The code coverage is 100%.

The total LOC of audited sources is 204.

AUDIT PROCESS

We started and finished the audit on January 27, 2025.

We inspected the materials provided for the audit. After that, we manually analyzed all the contracts within the scope of the audit and checked their logic. Among other things, we verified the following properties of the contracts:

- The correctness of the usage of ChainLink price feed;
- The correct access restrictions for the pausable and ownable functionality;
- The absence of discrepancies with the provided documentation.

We scanned the project with the following tools:

- Static analyzer Slither;
- Our plugin Slitherin with an extended set of rules;
- Semgrep rules for smart contracts. We also sent the results to the developers in the text file.

We ran tests and calculated the code coverage.

We combined in a private report all the verified issues we found during the manual audit or discovered by automated tools.

On January 28, 2025, we changed the status of the report to public since the developers decided to not make a codebase update.

MANUAL ANALYSIS

The contracts were completely manually analyzed, their logic was checked. Besides, the results of the automated analysis were manually verified. All the confirmed issues are described below.

Critical issues

Critical issues seriously endanger project security. They can lead to loss of funds or other catastrophic consequences. The contracts should not be deployed before these issues are fixed.

The audit showed no critical issues.

Medium severity issues

Medium severity issues can influence project operation in the current implementation. Bugs, loss of potential income, and other non-critical failures fall into this category, as well as potential problems related to incorrect system management. We highly recommend addressing them.

M01. The amount of sold tokens is not checked

According to the provided documentation:

> The presale volume is unlimited.

However, the total supply of the token in the **Token.sol** file is capped. Hence, there might be a situation when the amount of sold tokens is higher than the total supply.

M02. Missing price age check

The price for the native coin is received from the ChainLink by using the `latestRoundData` function. In the current implementation, there are no checks on the age of that price. We recommend ensuring that the used price is up to date.

M03. Not all contracts are initialized

The `initialize` function of the **TokenPresale** contract does not invoke the initializator of the **PausableUpgradeable** contract.

M04. Owner role

The project heavily relies on the `owner` role. It has the following powers:

- Pause and unpause the **TokenPresale** contract at any time;
- Set the price of the token at any time;
- Add or remove discount codes and change the discount value at any time;
- Add or remove payment tokens;
- Update the treasury address;
- Add an arbitrary amount of purchased tokens to any user.

Some scenarios can lead to undesirable consequences for the project and its users, e.g. if the owner's private key becomes compromised. We recommend designing contracts in a trustless manner or implementing proper key management, e.g., setting up a multisig.

Low severity issues

Low severity issues do not directly affect project operation. However, they might lead to various problems in future versions of the code. We recommend fixing them or explaining why the team has chosen a particular option.

L01. Bug in the tests

We could not run tests without adding an arbitrary address argument to `await AggregatorV3.deploy()` function call in the **TokenPresale.js**.

L02. Missing dependency

Token.sol file imports contracts from the `@openzeppelin/contracts` library. However, it is not added as a dependency in the **package.json**.

L03. Missing sequencer uptime check

According to ChainLink documentation:

> If you are reading data feeds on L2 networks, you must check the latest answer from the L2 Sequencer Uptime Feed to ensure that the data is accurate in the event of an L2 sequencer outage.

Consider referring to the following page for details.

Notes

N01. Discount code may be used multiple times

According to the developers, discount codes may be used multiple times. Note that, since transactions are not private, anyone is able to find the best discount code and use it.

N02. Price of stablecoins

The **TokenPresale** contract assumes that the price of all stablecoins is 1$, which is not the case. There might be the cases when this price difference might be important, for example in the case of stablecoin depeg.

This analysis was performed by Pessimistic:

Daria Korepanova, Senior Security Engineer
Pavel Kondratenkov, Senior Security Engineer
Irina Vikhareva, Project Manager
Alexander Seleznev, CEO

January 28, 2025