Smart Contract Audit Report

1. Overview

- **Project Name:** Larva
- **Contracts Audited:**
    - File: Token.sol
- **Code base: https://etherscan.io/address/0x9873E422C9cfFe275eF5eb3377E120E91f4d4630#code**
- **Network:** Etherscan

2. Executive Summary

This audit aims to assess the security of the smart contracts for the ICO protocol. We identified issues:

- High Risk
- Medium Risk
- Low Risk
- Informational / Gas Optimization Suggestions

All high and medium severity issues have been addressed / partially addressed / not addressed by the development team at the time of publishing this report.

3. Methodology

Our audit involved the following steps:

1. **Manual Code Review** – Analyzed logic for correctness, authorization, error handling, etc.
2. **Automated Testing** – Used static analysis tools such as Slither, MythX, or Hardhat plugins.
3. **Gas Usage Review** – Identified areas for optimization.
4. **Dependency Analysis** – Assessed third-party libraries (e.g., OpenZeppelin).

4. Risk Classification

Severity	Description
High	Critical vulnerabilities that can lead to loss.
Medium	Significant issues but not critical.
Low	Minor issues, often best practices.
Informational	No direct risk but useful suggestions.

5. Inheritance Graph

NA

6. Ownership Privileges

Token.sol

- The owner can set the uni swap pool address.
- The owner can manually burn his tokens.
- **Note** - This Audit report consists of a security analysis of the **Larva** smart contract. This analysis did not include functional testing (or unit testing) of the contract's logic. Moreover, we only audited one token contract for the **Larva** team. Other contracts associated with the project were not audited by our team. We recommend investors do their own research before investing.

7. Findings

7.1 High Severity

[H-01] Unsafe Usage of tx.origin.

- **Description:** Avoid using TX.origin for authorization, another contract can have a method that will call your contract (where the user has some funds for instance) and your contract will authorize that transaction as your address is in tx.origin
- **Mitigation:** It is recommended that you use msg.sender for authorization (if another contract calls your contract msg.sender will be the address of the contract and not the address of the user who called the contract.
- File: Token.sol
- Line: L374
- **Status:** Open

7.2 Medium Severity

7.3 Low Severity

7.4 Informational/Optimization Severity

[I-01] Remove unused functions.

- **Description:** Functions that are not used (dead-code).
- File: Token.sol
- Line: 120-123
- **Status:** Open

8. Conclusion

The smart contracts are generally well-written and follow modern best practices. After addressing the findings noted, the contracts should be considered safe for deployment, assuming no changes are made without further auditing.