Author: ChatGPT, referencing insights from Quantstamp audits

Date: 01 Nov 2024

Methodology

This audit was prompted to study the structure and best practices in Quantstamp audits to guide its analysis of the "861871" contract. Based on those standards, ChatGPT meticulously reviewed the contract code and reached the following conclusions regarding its security and compliance with ERC-20 norms.

Contract Information

Contract Address:
0x0A49A0af017Dbb19228b3519432B4790650fc03E
Token Symbol:
"861871"
Compiler Version:
Solidity 0.8.23
License:
UNLICENSE

Executive Summary

This audit provides a thorough analysis of the ERC-20 smart contract associated with token "861871" to determine its security and functionality. The audit covers its tokenomics, security controls, and post-deployment modifications by the contract owner, with a focus on assessing whether these modifications align with industry best practices. Notably, ownership has been renounced, ensuring no further administrative control by the original deployer.

Key Findings

- Critical Issues: None
- High-Risk Issues: None
- Medium-Risk Issues: None
- Low-Risk Issues: None
- Informational Findings: Minimal control retained by the '_taxWallet' over two low-impact functions.

The '861871' contract is a standard ERC-20 implementation with additional functionalities for transfer taxation, anti-bot protection, and liquidity handling. Key elements of the contract include:

- Initial Total Supply:

1,000,000,000 tokens, fully circulating.

- Decimals:

9

- Ownership:

Ownership has been renounced, and all liquidity provider (LP) tokens have been sent to a burn address, permanently locking the liquidity and eliminating any risk of a rug pull.

Core Functionality

Taxation Mechanism:

The contract allows for transfer tax to be applied but is currently set to zero.

Liquidity Pool Management:

Automatic transfer to liquidity pool with tokens burned to lock liquidity, though with zero tax, this function is inactive.

Anti-Bot Mechanisms:

List of known bots, disabling functionality for listed addresses.

Owner's Post-Deployment Transactions

1. Enable Trading:

The owner enabled trading by adding liquidity to Uniswap and setting the 'tradingOpen' variable to true.

2. Remove Transaction Limits:

The owner called 'removeLimits()' to eliminate transaction limits, increasing the maximum transaction and wallet size to the total supply.

3. Reduce Fees to Zero:

The owner adjusted the tax rate to zero for all transactions by setting the '_newFee' parameter to 0.

4. Renounce Ownership:

Ownership was renounced by transferring control to 'address(0)', preventing any further modifications to the contract.

5. Liquidity Burn:

The owner permanently burned liquidity by transferring all LP tokens to the burn address, ensuring they cannot be retrieved. This action effectively prevents any future manipulation of the liquidity pool.

Tax Wallet Control and Post-Renouncement Functionality

The `_taxWallet` retains minimal control with access to the following functions:

- `reduceFee(uint256 _newFee)` - Allows for tax adjustment but is currently locked at 0%.
- `manualSwap()` - Permits manual token swaps for ETH, though inapplicable due to zero accumulation of taxed tokens.

Risk Impact: These permissions do not impose any risk on token holders since the tax is fixed at zero, and ownership renouncement prevents further modifications.

Tokenomics

- Total Supply: 1,000,000,000
- Circulating Supply: Fully circulating
- Transaction Fees: Permanently set to zero.
- Liquidity Lock: LP tokens burned, securing the liquidity pool against tampering.

Security and Compliance

- ERC-20 Compliance: Adheres to ERC-20 standards for all basic functions.
- Reentrancy Guard: `lockTheSwap` modifier implemented to prevent reentrancy attacks.
- Ownership Renouncement: Ensures contract immutability by setting the owner to `address(0)`.

Anti-Bot Protections

- Bot List: Finalized upon ownership renouncement; includes a mapping of known bot addresses, disabling functions for listed accounts.
- Impact: Anti-bot measures ensure fairness, and no future adjustments can be made due to renouncement.

Conclusion

No critical vulnerabilities were found that could lead to loss of user funds or unauthorized access. The contract's design, combined with the owner's actions--including enabling trading, reducing fees to zero, removing limits, and renouncing ownership--provides a high level of security and trustworthiness. The liquidity remains locked, with no potential for a rug pull.