soken

Smart Contract Audit Report

CHIRPPAD

soken

1. Audit details

Token Name	**CHIRPPAD**
Contract address	**0xF2e244d4020C182e8E2C936d4055E3F0e578064F**
Contract URL	**https://basescan.org/address/0xF2e244d4020C182e8E2C936d4055E3F0e578064F#code**
Language	**Solidity**
Platform	**Base Network**
Date	**8 February 2025**

General Overview

The ChirpPad (CHPD) contract is an ERC20 token implementing the ERC20Permit extension from OpenZeppelin, allowing off-chain approvals. The contract mints a fixed supply of 1 billion CHPD tokens to the deployer's address, following a standard ERC20 structure.

The contract ensures compliance with Solidity 0.8.28, leveraging OpenZeppelin's security-enhanced libraries. However, potential risks include centralization concerns and a lack of an ownership transfer mechanism. This audit identifies key security issues, best practices, and recommendations for enhancing the contract's security and robustness.

1. Security Risks Identified

1.1 Centralized Token Supply

- **Issue**: The contract mints the entire token supply to the deployer's address.
- **Risk**: This centralizes control over all tokens, leading to concerns about misuse, manipulation, or a potential rug pull.
- **Recommendation**: Implement vesting schedules, time-locked wallets, or multi-signature governance to ensure fair and transparent distribution.

1.2 Lack of Ownership Transfer Function

- **Issue**: The contract does not implement an ownership transfer mechanism.
- **Risk**: If the deployer loses access to their wallet, there will be no way to upgrade or manage the contract, causing a potential loss of governance.
- **Recommendation**: Integrate Ownable.sol and provide a function to transfer ownership securely.

1.3 No Emergency Pause Function

- **Issue**: The contract does not include a pause mechanism.
- **Risk**: If an exploit or attack is detected, there is no way to halt transactions, potentially causing financial loss.
- **Recommendation**: Implement Pausable.sol to allow the owner to temporarily stop transactions in emergency situations.

1.4 No Rate-Limiting or Anti-Bot Measures

- **Issue**: There are no mechanisms to prevent bot trading or front-running attacks.
- **Risk**: This can lead to unfair trading conditions, where bots accumulate a large portion of the supply before the community.
- **Recommendation**: Introduce anti-bot measures, such as:

A cooldown period between transactions.
A maximum transaction limit per block.

2. Best Practices and Optimizations

2.1 Gas Optimization in ERC20 Transactions

- **Observation**: The contract follows OpenZeppelin standards but should be tested for gas efficiency on Layer 2 networks.
- **Recommendation**: Optimize ERC20 functions to reduce gas fees, particularly in transferFrom and approve functions.

2.2 Explicit Use of Visibility Modifiers

- Observation: Function visibility is explicitly declared, which is a good practice.
- Recommendation: Maintain explicit visibility (public, external, internal, private) for all functions.

2.3 Implement Reentrancy Guard for Future Upgrades

- **Observation**: The contract currently does not handle external calls, but future functionalities (like staking) may require security against reentrancy attacks.
- **Recommendation**: Use ReentrancyGuard.sol in any function dealing with external calls or withdrawals.

2.4 Missing Events for Key Transactions

- **Observation**: The contract lacks event logging for ownership transfers and approvals.
- **Recommendation**: Add events for ownership changes, approvals, and emergency actions to improve traceability.

3. Feature Recommendations

3.1 Decentralized Governance Model

Suggestion: Implement DAO-based governance to allow token holders to vote on decisions rather than relying on a single deployer address.

3.2 Implementing a Burn Mechanism

Suggestion: Consider a burn function to enable deflationary economics. This can increase token scarcity and value over time.

3.3 Time-Locking Owner Privileges

Suggestion: Introduce a time-lock contract for major administrative actions, such as:
- Token recovery
- Ownership transfer
- Supply adjustments

This prevents sudden malicious changes and allows users to react.

3. Feature Recommendations

3.1 Decentralized Governance Model

Suggestion: Implement DAO-based governance to allow token holders to vote on decisions rather than relying on a single deployer address.

3.2 Implementing a Burn Mechanism

Suggestion: Consider a burn function to enable deflationary economics. This can increase token scarcity and value over time.

3.3 Time-Locking Owner Privileges

Suggestion: Introduce a time-lock contract for major administrative actions, such as:
- Token recovery
- Ownership transfer
- Supply adjustments

This prevents sudden malicious changes and allows users to react.

4. Severity Matrix

Centralized Token Supply

Severity: High
Likelihood: Medium
Impact: High
Priority: Critical

Lack of Ownership Transfer Function

Severity: Medium
Likelihood: Medium
Impact: High
Priority: High

No Emergency Pause Function

Severity: Medium
Likelihood: Medium
Impact: Medium
Priority: High

No Rate-Limiting/Anti-Bot Measures

Severity: Medium
Likelihood: High
Impact: Medium
Priority: High

Gas Optimization for ERC20 Transactions

Severity: Low
Likelihood: Medium
Impact: Low
Priority: Medium

Missing Event Logging for Key Transactions

Severity: Low
Likelihood: Low
Impact: Medium
Priority: Medium

5. Conclusion

The ChirpPad (CHPD) contract is a well-implemented ERC20 token that leverages OpenZeppelin's security standards. However, centralization risks, missing ownership transfer functionality, and the absence of emergency controls are notable concerns.

By implementing the recommended improvements, such as vesting schedules, governance models, and security enhancements, the contract can become more secure, transparent, and decentralized.

Final Recommendation: ☐ Pass with Critical Fixes Required

Ownership transfer and emergency pause functionality must be added before deployment.
Consider implementing decentralized governance to ensure long-term stability.

SOKEN.IO

WEBSITE: WWW.SOKEN.IO

TELEGRAM: @SOKEN_SUPPORT

X (TWITTER): @SOKEN_TEAM