Smart Contract Security Audit Report

Project: ARSXEngine

Auditor: ChatGPT

Date: July 2025

This report presents a security and code quality review of the ARSXEngine smart contract. The goal is to ensure robustness, security, and maintainability, while highlighting potential improvements following best practices.

General Strengths

- Use of immutable for ARSX token reference improves security and gas efficiency.
- Clear modular design separating collateral and stablecoin logic.
- Proper use of nonReentrant guards to prevent reentrancy attacks.
- Good usage of custom errors to save gas and improve clarity.
- Explicit error handling and descriptive revert messages.

Security Observations & Recommendations

1. Health Factor Check: Current implementation requires the user's health factor to strictly improve after liquidation. Consider allowing liquidation as long as the health factor does not worsen, enabling partial liquidations and reducing protocol risk.
2. Collateral Balance Underflow: Add explicit checks to prevent underflow when redeeming collateral balances.
3. ERC20 Allowance Handling: Provide clearer error messages when transferFrom fails due to insufficient allowance for better UX.
4. Oracle Dependency: Ensure price freshness checks are correctly enforced in OracleLib to prevent stale data manipulation.
5. Liquidation Incentive Configuration: Consider making the liquidation bonus adjustable via governance or configurable, to react to extreme market conditions.
6. Reentrancy: While using nonReentrant, continue to follow Checks-Effects-Interactions pattern explicitly to improve clarity and defense-in-depth.

Code Quality & Best Practices

- Use explicit uint256 type instead of uint to improve consistency and clarity.
- Move all constants to the top and group them together for easier auditing.
- Split complex functions like liquidate into smaller, single-responsibility internal functions.
- Add events for minting and burning ARSX to improve transparency and off-chain indexing.
- Use consistent naming conventions, e.g., prefer 'collateralToken' instead of 'collateral' to clarify variable roles.
- Expand function-level docstrings to include @dev, @param, and @return details.

Overall Summary

The ARSXEngine contract is designed with solid security fundamentals and good modular structure.

We recommend implementing the above improvements to further strengthen protocol resilience, enhance maintainability, and improve developer and user experience.

If desired, we can also provide a refactored contract incorporating these recommendations or a detailed threat model diagram.