Smart Contract Security Audit Report

Executive Summary: FibonacciBalance.sol

Audit Date: 2025-12-19 14:12:52
Total Issues Found: 11

Breakdown:
- Critical: 1
- High: 1
- Medium: 1
- Low: 8

Smart Contract Security Audit Report

Detailed Vulnerability Findings

Issue: [Logic] Unprotected Function

Severity: Critical

Description: The fallback function allows users to call any function in the FibonacciLib contract, including setStart, which can be used to manipulate the calculatedFibNumber and potentially drain the contract's funds.

Remediation: Add proper access control to the fallback function to restrict which functions can be called.

Issue: [Logic] Reentrancy Vulnerability

Severity: High

Description: The withdraw function uses the transfer function to send ether to the user, which can lead to a reentrancy attack if the user's contract reenters the withdraw function.

Remediation: Use the Checks-Effects-Interactions pattern to prevent reentrancy attacks.

Issue: [Logic] Unsecured Delegatecall

Severity: Medium

Description: The withdraw function uses delegatecall to call the setFibonacci function in the FibonacciLib contract, which can lead to unintended behavior if the FibonacciLib contract is not properly secured.

Remediation: Ensure that the FibonacciLib contract is properly secured and that the setFibonacci function is not vulnerable to manipulation.

Issue: [Automated] arbitrary-send-eth

Severity: Low

Description: The contract sends Ether to an arbitrary user based on the calculated Fibonacci number, which could lead to unintended fund transfers.

Remediation: Implement proper access control and validation for the withdrawal function to prevent unauthorized or unintended transfers.

Issue: [Automated] controlled-delegatecall

Severity: Low

Description: The contract uses delegatecall to a user-controlled function ID, which could lead to reentrancy attacks or other security vulnerabilities.

Remediation: Use a secure and validated function ID for delegatecall, and consider implementing reentrancy protection mechanisms.

Issue: [Automated] missing-zero-check

Severity: Low

Description: The contract lacks a zero-check for the _fibonacciLibrary address, which could lead to unintended behavior or errors if a zero address is provided.

Remediation: Add a zero-check for the _fibonacciLibrary address to prevent potential issues.

Issue: [Automated] deprecated-standards

Severity: Low

Description: The usage of 'sha3()' should be replaced with 'keccak256()'.

Remediation: Replace 'sha3()' with 'keccak256()'. For example: bytes4 constant fibSig = bytes4(keccak256("setFibonacci(uint256)"));

Issue: [Automated] solc-version

Severity: Low

Description: The Solidity version used is outdated and may contain known issues.

Remediation: Update the Solidity version to a more recent one, if possible.

Issue: [Automated] low-level-calls

Severity: Low

Description: The use of 'delegatecall' can pose a risk if not properly validated.

Remediation: Ensure that the 'delegatecall' is properly validated and secured.

Issue: [Automated] low-level-calls

Severity: Low

Description: The contract uses a low-level delegatecall to invoke a function from the FibonacciLib library. This could potentially lead to unintended behavior if the library is not properly secured.

Remediation: Use a secure library and validate the input data before invoking the delegatecall.

Issue: [Logic] Potential Gas Limit Issue

Severity: Low

Description: The fibonacci function in the FibonacciLib contract uses recursion, which can lead to a gas limit issue if the input is too large.

Remediation: Consider using an iterative approach instead of recursion to calculate the fibonacci number.
