Hurricane Finance
Smart Contract Audit

The BlockChain Auditor

Prepared by: Roger Blackstone

Version: 2
Date: Dec. 7, 2020

Summary of Findings

This document expresses all security concerns of the Hurricane Finance Smart Contracts as expressed by Roger Blackstone. I took care to attempt to find as many ways to improve the security, code efficiency, best practices, and overall function of the smart contracts.

Contract Status: Deployed at

0x2E8d989E2B2b2C55ba330b3F672d872256aD12e2

1 Critical Issues found in the categoryFive.sol were resolved.

0 Medium Issues were found.

0 Low Issues were found.

19 Informational Issues were found in categoryFive.sol were taken into consideration.

Code Coverage

Hurricane Finance	Industry Standard
~95.0%	95.0%

This audit should be seen as one step in the development process with the intent of raising awareness on the meticulous work involved in secure development and making no material statements or guarantees to the operational state of the smart contract(s) once they are deployed. This document is not an endorsement of the reliability or effectiveness of the smart contracts. This is an assessment of the smart contract logic, implementation, and best practices. I cannot take responsibility for any potential consequences of the deployment or use of the smart contract(s) related to the audit.

Table of Contents

Audit Methodology & Techniques

The BlockChain Auditor has the following auditing process:

1. Our audits include
   a. Review of the specifications, source code, and instructions provided to the BlockchainAuditors to clearly identify the desired functionality of the smart contract(s).
   b. Manual line by line review of contract code to spot potential vulnerabilities.
   c. Identification of deviations between desired functionality expressed to the BlockchainAuditors and what the smart contract(s) are doing.
2. Automated static and symbolic analysis, as well as verifying testing coverage using the provided test suite.
   a. Automated static and symbolic analysis help determine what inputs cause each part of the smart contract to execute. Analysis of how much of the code base is tested and comparison to industry standard.
3. Examination of smart contracts and development process as a whole, ensuring best practices are followed, allowing improved efficiency and security based on established industry and academic practices.
4. Specific, itemized, and actionable recommendations to assist in securing the smart contract(s) in question.

Contract Checklist

CategoryFive.sol

Contract Vulnerability	
Integer Overflow	Pass
Race Condition	Pass
Denial of Service	Pass
Logical Vulnerability	Pass
Hardcoded Address	Pass
Function Input Parameter Check	Pass
Function Access Control Check	Pass
Random Number Generation	N/A
Random Number Use	N/A
Contract Specification	
Solidity Compiler Version	Pass
Event Use	Pass
Fallback Function Use	Pass
Constructor Use	Pass
Function Visibility Declaration	Pass
Variable Storage Declaration	Pass
Deprecated Keyword Use	Pass
ERC20/223 Standard	Pass
ERC721 Standard	N/A
Business Risk	
Able to Arbitratrily Create Token	Pass
Able to Arbitrarily Destroy Token	Pass
Can Suspend Transactions	N/A
Short Address Attack	Pass
Gas Optimization	
assert()/require()/revert() misused	Pass
Loop Optimization	Pass
Storage Optimization	Pass

Notable Concerns & Suggestions

Overall Thoughts

This is a follow up on the previous version of the audit verifying that no malicious changes were done after the first audit. This analysis was done on the smart contracts deployed at address 0x2E8d989E2B2b2C55ba330b3F672d872256aD12e2. You can compare the file fingerprints on this contract by calculating the MD5 checksums of the deployed files.

Since the last audit the team has further modularized their code by extracting fee calculation functions into their own library inside of FeeHelpers.sol.

Optimizations have been made wherever possible.

The Hurricane Finance team finalized their implementation of community governance. The code also adheres to what is written in the litepaper.

The team also has plenty of unit tests that verifies that the codebase works as expected.

As before, the protocol is well documented inside and outside of the code. I did not find any discrepencies with the code specification and the implementation; in fact, it was executed so well along the specifications that it will be remembered for its adherence to code integrity.

Appendix A

File Fingerprints

Cane.sol	34f577c7781b710704d59d39ea5880cc
Hugo.sol	70fd4927597ce4f9fa77bbb11ef8a51c
CategoryFive.sol	eae6a5e80fcf8625c443ca5448d6d861
lib/Address.sol	f6ef8d03426fc051eaf3e86caa776669
lib/Context.sol	c345d83fb6462ef86d81d3453aab75ba
lib/ERC20.sol	92c13c1bf9df9818c9902d7008e20718
lib/FeeHelpers.sol	0438e7f7913000d94910333f44d3e591
lib/Math.sol	2097525639e8f8aa49fa57265e0278ef
lib/Ownable.sol	accee08ca980e3696d6b1bb4271bf4bf
lib/ReentrancyGuard.sol	b0ff74ed8c7dcfa37436c8be87c79750
lib/SafeERC20.sol	a807c2c10ab8d513111e4fbf79b67be8
lib/SafeMath.sol	7b036fb13cf1240f03c4cb9b9f60411c
interfaces/IERC20.sol	2f67604b284cd200fccfe111b02920c6
interfaces/IUniswap.sol	94adf9d4d2ad7c4fd123caef8a5ba008
interfaces/IWETH.sol	2adbecc444dd30adfdc9d6425fded6cc
test/categoryFive.js	69f208b3533fe04bc6fb2e31014492d4

The Blockchain Auditor is honored to have had the opportunity to partner with Hurricane Finance and help provide enhanced security and efficiency for the Hurricane Finance platform and the crypto community as a whole.

The Hurricane Finance staking and farming protocol is excellently implemented as stated in their litepaper and as such is a prime example of code integrity.

The BlockChain Auditor

- Roger Blackstone