Smart contract security audit Nerveflux

v.1.2

Table of Contents

1.0 Introduction

1.1 Project engagement

During September of 2021, Nerveflux engaged CTDSec to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. Nerveflux provided CTDSec with access to their code repository and whitepaper.

1.2 Disclaimer

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that's able to withstand the network's fast-paced and rapidly changing environment, we at CTDSec recommend that Nerveflux team put in place a bug bounty program to encourage further and active analysis of the smart contract.

2.0 Coverage

2.1 Target Code and Revision

For this audit, we performed research, investigation, and review of the Nerveflux contract followed by issue reporting, along with mitigation and remediation instructions outlined in this report. The following code files are considered in-scope for the review:

Source:

nerveflux_n.txt [SHA256] -

2eccb77edf653d28c44eedbd6170d91c460e9423fae4225f56d35a889bf7ea0e

Updated version:

nerveflux_n (1a).txt [SHA256] -

7ece7cbaef8146bfe0a0b9490e16008051dbe33be40cd39c4aa4cf7e1bafa5ae

2.2 Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices.

№	Issue description.	Checking status
1	Compiler warnings.	PASSED
2	Race conditions and Reentrancy. Cross-function race conditions.	PASSED
3	Possible delays in data delivery.	PASSED
4	Oracle calls.	PASSED
5	Front running.	PASSED
6	Timestamp dependence.	PASSED
7	Integer Overflow and Underflow.	PASSED
8	DoS with Revert.	PASSED
9	DoS with block gas limit.	PASSED
10	Methods execution permissions.	PASSED
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	PASSED
12	The impact of the exchange rate on the logic.	PASSED
13	Private user data leaks.	PASSED
14	Malicious Event log.	PASSED
15	Scoping and Declarations.	PASSED
16	Uninitialized storage pointers.	PASSED
17	Arithmetic accuracy.	PASSED

18	Design Logic.	SOLVED ISSUES
19	Cross-function race conditions.	PASSED
20	Safe Zeppelin module.	PASSED
21	Fallback function security.	PASSED
22	Overpowered functions / Owner privileges	PASSED

3.0 Security Issues

3.1 High severity issues [0]

1.Unassigned wallets

Charitywalelt and staking wallets are unassigned.

```
address public _charityWalletAddress = address(0x123); //replace charity wallet here
address public _stakingWalletAddress = address(0x1234); // replace staking wallet here
```

Recommendation:

Assign the variables to the wallets.

Update:

Wallets are replaced and fixed:

```
1040    address public _charityWalletAddress = address(0x3174f90Fc5871c280FD72a4149Cb4383c6c0DC32); //
    Charity wallet
1041    address public _stakingWalletAddress = address(0x872e61a8DBb39Bd639CB810e4D4a44463aB6f039); //
    staking wallet
```

3.2 Medium severity issues [0]

No medium severity issues found.

3.3 Low severity issues [0]

No low severity issues found.

4.0 Owner privileges

- Owner can blacklist address
- Owner can exclude address from fees/max transaction amount
- Onwer can claim stuck tokens and Dust BNB
- Owner can change charity wallet
- Owner can change staking wallet
- Owner can change liquidity fee
- Owner can change charity fee
- Owner can change burn fee
- Owner can set automatedmarketpair
- Owner can set a minimum tokens amount for swap

5.0 Summary of the audit

Contract has no issues and it’s safe to be deployed.