AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

NeiroGrok

DATED : 20 Oct 24'

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

Function: Launch

Status: Open

Overview:

The Launch function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function Launch( bool status) external onlyOwner {
    if (!status){
      starBlock = 0;
    }else{
      starBlock = block.number;
    }
}
```

Suggestion:

To reduce centralization and potential manipulation, consider one of the following approaches:

•Automatically enable trading after a specified condition, such as the completion of a presale, is met.

•If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

MANUAL TESTING

Centralization – Buy and Sell Fees.

Severity: High

Function: setBuy and setSell.

Status: Open

Overview:

The owner can set the buy and sell fees up to 100%, which is not recommended.

```
function setBuy(uint256 newFundFee, uint256 newLpFee) public onlyOwner{
    _buyFundFee = newFundFee;
    _buyLPFee = newLpFee;
}

function setSell(uint256 newFundFee, uint256 newLpFee) public onlyOwner{
    _sellFundFee = newFundFee;
    _sellLPFee = newLpFee;
}
```

Suggestion:

It is recommended that no fees in the contract should be more than 25% of the contract.

AUDIT SUMMARY

Project name – NeiroGrok

Date: 20 Oct, 2024

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: **High-risk major flag**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	2	2	3	1
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
A line by line code review has been performed by audit ace team.

2- BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:

https://testnet.bscscan.com/address/0x0cdb77de16b144d62a902e7feee9cc3c18df4b01#code

Token Information

Token Address:
0x4756FE6422D372Cb46e71D7Aa6849BaD6DA62f16

Name: NeiroGrok

Symbol: NeiroGrok

Decimals: 9

Network: Binance Smart Chain

Token Type: BEP-20

Owner: 0x20A6F64a56b2a14c0B1FFa5Be4F8F7D014EAfBa4

Deployer: 0x20A6F64a56b2a14c0B1FFa5Be4F8F7D014EAfBa4

Token Supply: 420000000000000000

Checksum: BEde641126e217b2b455d49e77fc41h23

Testnet:
https://testnet.bscscan.com/address/0x0cdb77de16b144d62a902e7feee9cc3c18df4b01#code

TOKEN OVERVIEW

Buy Fee: 50-100%

Sell Fee: 50-100%

Transfer Fee: 0-0%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] Using throw

INHERITANCE TREE

POINTS TO NOTE

- The owner can transfer ownership.
- The owner can renounce ownership.
- The owner can set the fund address.
- The owner can set the Buy and Sell fee to more than 100%.
- The owner can start trading.
- The owner whitelist addresses.
- The owner can enable airdrop.
- The owner can set a swap pair list address.

STATIC ANALYSIS

```
INFO:Detectors:
AbsToken.transferToAddressETH(address,uint256) (NeiroGrok.sol#362-364) sends eth to arbitrary user
        Dangerous calls:
        - recipient.transfer(amount) (NeiroGrok.sol#363)
AbsToken.claimBalance() (NeiroGrok.sol#399-401) sends eth to arbitrary user
        Dangerous calls:
        - address(fundAddress).transfer(address(this).balance) (NeiroGrok.sol#400)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations
```

```
INFO:Detectors:
AbsToken._tokenTransfer(address,address,uint256,bool,bool).feeAmount (NeiroGrok.sol#295) is a local variable never initialized
AbsToken._transfer(address,address,uint256).isSell (NeiroGrok.sol#251) is a local variable never initialized
AbsToken._transfer(address,address,uint256).takeFee (NeiroGrok.sol#250) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
TokenDistributor.constructor(address) (NeiroGrok.sol#93-95) ignores return value by IERC20(token).approve(msg.sender,uint256(~ uint25
6(0))) (NeiroGrok.sol#94)
AbsToken.constructor(address,string,string,uint8,uint256,address) (NeiroGrok.sol#135-173) ignores return value by IERC20(WBNBAddress)
.approve(address(swapRouter),MAX) (NeiroGrok.sol#146)
AbsToken.swapTokenForFund(uint256,uint256) (NeiroGrok.sol#322-360) ignores return value by _swapRouter.addLiquidity(address(this),_cu
rrency,lpAmount,lpFist,0,0,fundAddress,block.timestamp) (NeiroGrok.sol#355-357)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
AbsToken.allowance(address,address).owner (NeiroGrok.sol#200) shadows:
        - Ownable.owner() (NeiroGrok.sol#71-73) (function)
AbsToken._approve(address,address,uint256).owner (NeiroGrok.sol#227) shadows:
        - Ownable.owner() (NeiroGrok.sol#71-73) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
AbsToken.setFundAddress(address).addr (NeiroGrok.sol#382) lacks a zero-check on :
                - fundAddress = addr (NeiroGrok.sol#383)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
```

```
INFO:Detectors:
NeiroGrok.constructor() (NeiroGrok.sol#423-431) uses literals with too many digits:
        - AbsToken(address(0xD99D1c33F9fC3444f8101754aBC46c52416550D1),NeiroGrok,NeiroGrok,9,420000000000000000,address(0xaC4919273d9
BcfE68bbE4A6D36952A17E2FC75d1)) (NeiroGrok.sol#423-430)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
AbsToken._currency (NeiroGrok.sol#113) should be immutable
AbsToken._mainPair (NeiroGrok.sol#127) should be immutable
AbsToken._swapRouter (NeiroGrok.sol#112) should be immutable
AbsToken._tokenDistributor (NeiroGrok.sol#118) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:NeiroGrok.sol analyzed (8 contracts with 94 detectors), 46 result(s) found
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

1- Set Buy Fee (passed):

https://testnet.bscscan.com/tx/0x38eee7233e8c3ec58a5d0854089440ccf991c181f24fb225c50742552a2e2407

2- Set Sell Fee (passed):

https://testnet.bscscan.com/tx/0xc3c4dfce55a7e9205491758a5f0747fd658299f5428ff5e6d0e02637827bf394

3- Launch (passed):

https://testnet.bscscan.com/tx/0x947d3b1507a1dd0a5c3e6cca771b5f0095cea7d4a0656add9436cdd8af8d81a3

4- Set Fund Address (passed):

https://testnet.bscscan.com/tx/0xa99336e048cee26189ca5ca29c34815efb5eb2b3d12c11525fa6dcd711a9f11c

CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	2
Medium-Risk	2
Low-Risk	3
Gas Optimization / Suggestions	1

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

Function: Launch

Status: Open

Overview:

The Launch function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function Launch( bool status) external onlyOwner {
    if (!status){
      starBlock = 0;
    }else{
      starBlock = block.number;
    }
}
```

Suggestion:

To reduce centralization and potential manipulation, consider one of the following approaches:

•Automatically enable trading after a specified condition, such as the completion of a presale, is met.

•If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

MANUAL TESTING

Centralization – Buy and Sell Fees.

Severity: High

Function: setBuy and setSell.

Status: Open

Overview:

The owner can set the buy and sell fees up to 100%, which is not recommended.

```
function setBuy(uint256 newFundFee, uint256 newLpFee) public onlyOwner{
    _buyFundFee = newFundFee;
    _buyLPFee = newLpFee;
}

function setSell(uint256 newFundFee, uint256 newLpFee) public onlyOwner{
    _sellFundFee = newFundFee;
    _sellLPFee = newLpFee;
}
```

Suggestion:

It is recommended that no fees in the contract should be more than 25% of the contract.

MANUAL TESTING

Centralization – Liquidity is added to EOA.

Severity: Medium

function: add liquidity

Status: Open

Overview:

Liquidity is added to EOA. It may be drained by the fundAddress.

```
{
        try _swapRouter.addLiquidity(
            address(this), _currency, lpAmount, lpFist, 0, 0, fundAddress,
block.timestamp
        ) {} catch { emit Failed_addLiquidity(); }
    }
```

Suggestion:

It is suggested that the address should be a contract address or a dead address.

MANUAL TESTING

Centralization – Missing Require Check(HoneyPot)

Severity: Medium

Function: setFundAddress

Status: Open

Overview:

The owner can set any arbitrary address including zero address in FundAddress. This is not recommended because if the owner sets the address to the contract address, then the ETH will not be sent to that address and the transaction will fail and this will lead to a potential honeypot in the contract.

```
function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _isExcludeFromFee[addr] = true;
}
```

Suggestion:

It is recommended that the address should not be able to set as a contract address.

MANUAL TESTING

Centralization – Missing Events

Severity: Low

Subject: Missing Events

Status: Open

Overview:

They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _isExcludeFromFee[addr] = true;
}
 function setBuy(uint256 newFundFee, uint256 newLpFee) public onlyOwner{
    _buyFundFee = newFundFee;
    _buyLPFee = newLpFee;
}

function setSell(uint256 newFundFee, uint256 newLpFee) public onlyOwner{
    _sellFundFee = newFundFee;
    _sellLPFee = newLpFee;
}
function Launch( bool status) external onlyOwner {
    if (!status){
       starBlock = 0;
    }else{
       starBlock = block.number;
    }
}
function setAirDropEnable(bool status) public onlyOwner{
    airdropEnable = status;
}
function setSwapPairList(address addr, bool enable) external onlyOwner {
    _swapPairList[addr] = enable;
}
```

MANUAL TESTING

```
function multiWLs(address[] calldata addresses, bool value) public onlyOwner{
    require(addresses.length < 201);
    for (uint256 i; i < addresses.length; ++i) {
        _isExcludeFromFee[addresses[i]] = value;
    }
}
```

MANUAL TESTING

Centralization – Missing Zero Address

Severity: Low

Status: Open

Overview:
functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.

```
function setFundAddress(address addr) external onlyOwner {
    fundAddress = addr;
    _isExcludeFromFee[addr] = true;
}
```

Suggestion:
It is suggested that the address should not be zero or dead.

MANUAL TESTING

Centralization – Local Variable Shadowing

Severity: Low

Status: Open

Function: _approve and allowance

Overview:

```
function _approve(address owner, address spender, uint256 amount) private {
    _allowances[owner][spender] = amount;
    emit Approval(owner, spender, amount);
}
function allowance(address owner, address spender) public view override
returns (uint256) {
    return _allowances[owner][spender];
}
```

Suggestion:

Rename the local variable that shadows another component.

MANUAL TESTING

Optimization

Severity: Informational

Subject: Old Pragma Solidity version

Status: Open

Overview:

It is considered best practice to pick the latest compiler version and stick with it. With a floating pragma, contracts may accidentally be deployed using an outdated.

pragma solidity ^0.8.18;

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace