AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Grok X

DATED : 25 Nov 23'

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

function: OpenTrading

Status: Open

Overview:

The OpenTrading 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 openTrading() external onlyOwner {
    tradingOpen = true;
 }
```

Suggestion

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

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

2.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

AUDIT SUMMARY

Project name – Grok X

Date: 25 Nov, 2023

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

Audit Status: **Passed with high risk**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	0	1	2
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/0x356304f7c4178a5948c005d29cab1cd028176c71#code

Token Information

Token Address:
0x2388821b40F3Ab780F09e97b42b7b577d37A6d5E

Name: Grok X

Symbol: GROKX

Decimals: 18

Network: Etherscan

Token Type: ERC20

Owner:
0x176346ca8f88f91E3bfC54E57a52223Db8C94796

Deployer:
0x176346ca8f88f91E3bfC54E57a52223Db8C94796

Token Supply: 100000000000000000000000000

Checksum: adda898a610f626a9531512ab1ada887

Testnet:
https://testnet.bscscan.com/address/0x356304f7c4178a5948c005d29cab1cd028176c71#code

TOKEN OVERVIEW

Buy Fee: 0-0%

Sell Fee: 0-0%

Transfer Fee: 0-0%

Fee Privilege: --

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

Other Privileges:

-Whitelist to transfer without enabling trades

- Enabling trades

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

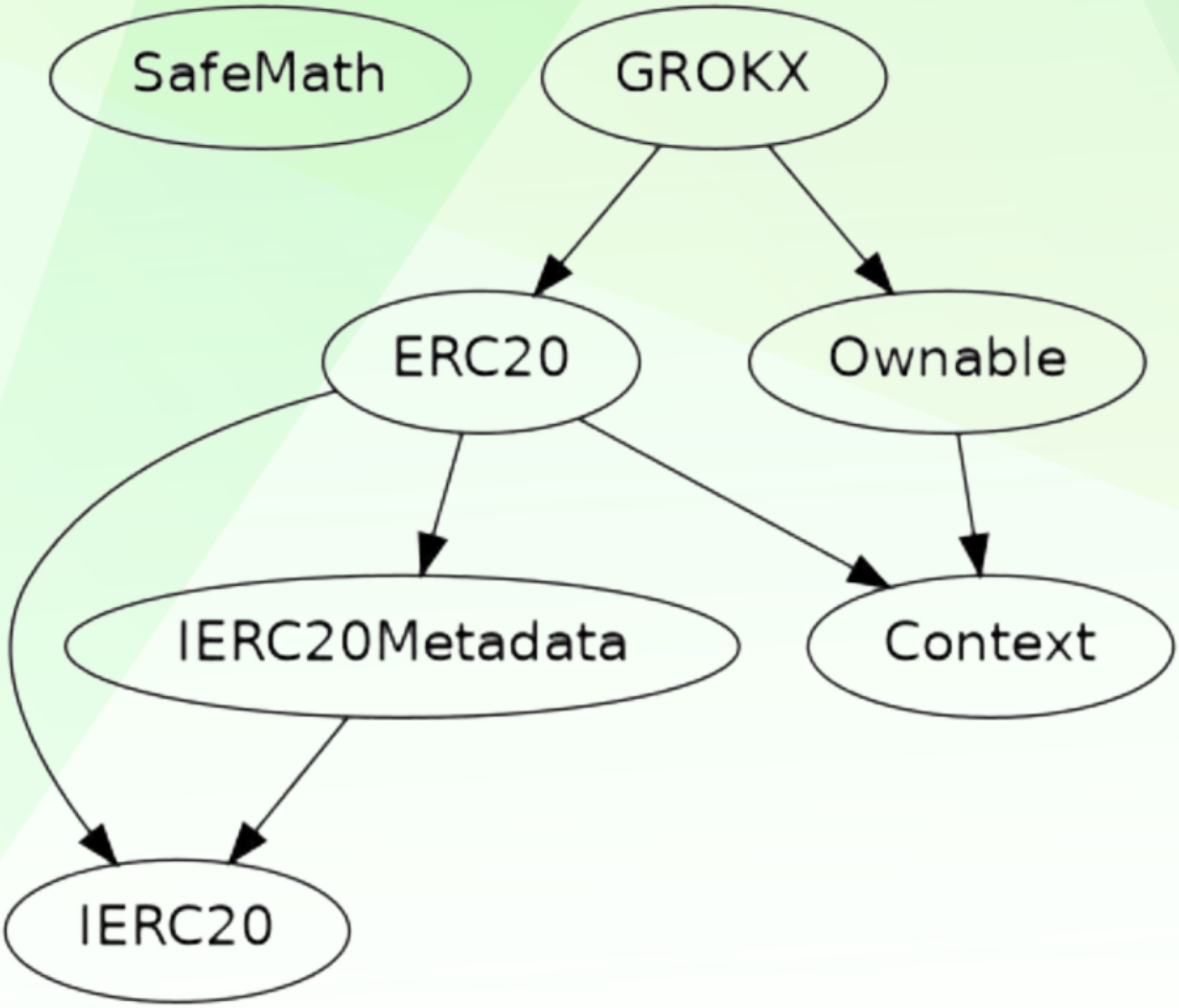
CLASSIFICATION OF RISK

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	1
Medium-Risk	0
Low-Risk	1
Gas Optimization / Suggestions	2

INHERITANCE TREE



STATIC ANALYSIS

```
INFO:Detectors:
Contract locking ether found:
        Contract GROKX (GROKX.sol#776-832) has payable functions:
         - GROKX.receive() (GROKX.sol#788)
        But does not have a function to withdraw the ether
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#contracts-that-lock-ether
INFO:Detectors:
GROKX.constructor().totalSupply (GROKX.sol#783) shadows:
        - ERC20.totalSupply() (GROKX.sol#409-411) (function)
        - IERC20.totalSupply() (GROKX.sol#242) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Context._msgData() (GROKX.sol#345-347) is never used and should be removed
ERC20._burn(address,uint256) (GROKX.sol#620-635) is never used and should be removed
SafeMath.add(uint256,uint256) (GROKX.sol#103-105) is never used and should be removed
SafeMath.div(uint256,uint256) (GROKX.sol#145-147) is never used and should be removed
SafeMath.div(uint256,uint256,string) (GROKX.sol#200-209) is never used and should be removed
SafeMath.mod(uint256,uint256) (GROKX.sol#160-162) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (GROKX.sol#226-235) is never used and should be removed
SafeMath.mul(uint256,uint256) (GROKX.sol#131-133) is never used and should be removed
SafeMath.sub(uint256,uint256) (GROKX.sol#117-119) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (GROKX.sol#177-186) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (GROKX.sol#17-26) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (GROKX.sol#68-76) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (GROKX.sol#83-91) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (GROKX.sol#48-61) is never used and should be removed
SafeMath.trySub(uint256,uint256) (GROKX.sol#33-41) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version>=0.8.19 (GROKX.sol#8) necessitates a version too recent to be trusted. Consider deploying with 0.8.18.
solc-0.8.22 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Parameter GROKX.setPreLaunchAddress(address,bool)._address (GROKX.sol#796) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
GROKX.constructor() (GROKX.sol#782-786) uses literals with too many digits:
        - totalSupply = 1000000000 * 10 ** 18 (GROKX.sol#783)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Slither:GROKX.sol analyzed (7 contracts with 93 detectors), 21 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- Approve (passed):

https://testnet.bscscan.com/tx/0x41db8b02cf3208352cddab6e33efbf7b2a8846f92919d96c1f4bdb252cb6578a

2- Increase Allowance (passed):

https://testnet.bscscan.com/tx/0x475c4742cd1b8f0793994e534df08d4dfccba9db807bad02dcb4eaa0330acb6d

3- Decrease Allowance (passed):

https://testnet.bscscan.com/tx/0xeb76fc786b9447661ab7fcf4c1674368c00a1b8010297e7f645a1b49cf5af321

4- Open Trading (passed):

https://testnet.bscscan.com/tx/0xdf6de50fbe0e234c976a0017403c224901d81cceece03e8ffdc77e5943ca3223

5- Set Pre Launch Address (passed):

https://testnet.bscscan.com/tx/0x4d95cd8a68b5189dea826cd0e0c7a680887074ef44eac3c06f9ead8aca51eb44

6- Transfer (passed):

https://testnet.bscscan.com/tx/0x71aa10e28f581a703428fc523dc100d12c39eea98793d5edf81511607d609043

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

function: OpenTrading

Status: Open

Overview:

The OpenTrading 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 openTrading() external onlyOwner {
    tradingOpen = true;
 }
```

Suggestion

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

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

2.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

Optimization

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 openTrading() external onlyOwner {
    tradingOpen = true;
 }

 function setPreLaunchAddress(
    address _address,
    bool state
 ) external onlyOwner {
    preLaunchAddress[_address] = state;
 }
```

MANUAL TESTING

Optimization

Severity: Informational

subject: Remove Safe Math

Status: Open

Overview:

compiler version above 0.8.0 has the ability to control arithmetic overflow/underflow, It is recommended to remove the unwanted code in order to avoid high gas fees.

MANUAL TESTING

Optimization

Severity: Informational

subject: floating Pragma Solidity version

Status: Open

Overview:

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

pragma solidity >=0.8.19;

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