AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

ETF

DATED : 21 October 23'

MANUAL TESTING

Centralization – Enabling Trades
Severity: High
function: startTrading
Status: Open

Overview:
The startTrading 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 startTrading() 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 – ETF

Date: 21 October 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	0	0
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/0x77d7635fae1d1c139a04de6e5e68f695a3eb97c7

Token Information

Token Address :
0x2FcBD5a6eb694d573D280664393681cB52b9a98b

Name: ETF

Symbol: ETF

Decimals: 18

Network: Ethereum

Token Type: ERC20

Owner: 0xd84509573bb190e5F7E543f866C0857501D7c880

Deployer:
0xd84509573bb190e5F7E543f866C0857501D7c880

Token Supply: 21,000,000

Checksum:
1666029b29a5f1ae543a23971ebc1e066fc0f1b5

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/0x77d7635fae1d1c139a04de6e5e68f695a3eb97c7

TOKEN OVERVIEW

buy fee: 0%

Sell fee: 0%

transfer fee: 0%

Fee Privilege: No fees

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

Other Privileges:

- Initial distribution of the tokens
- 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

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

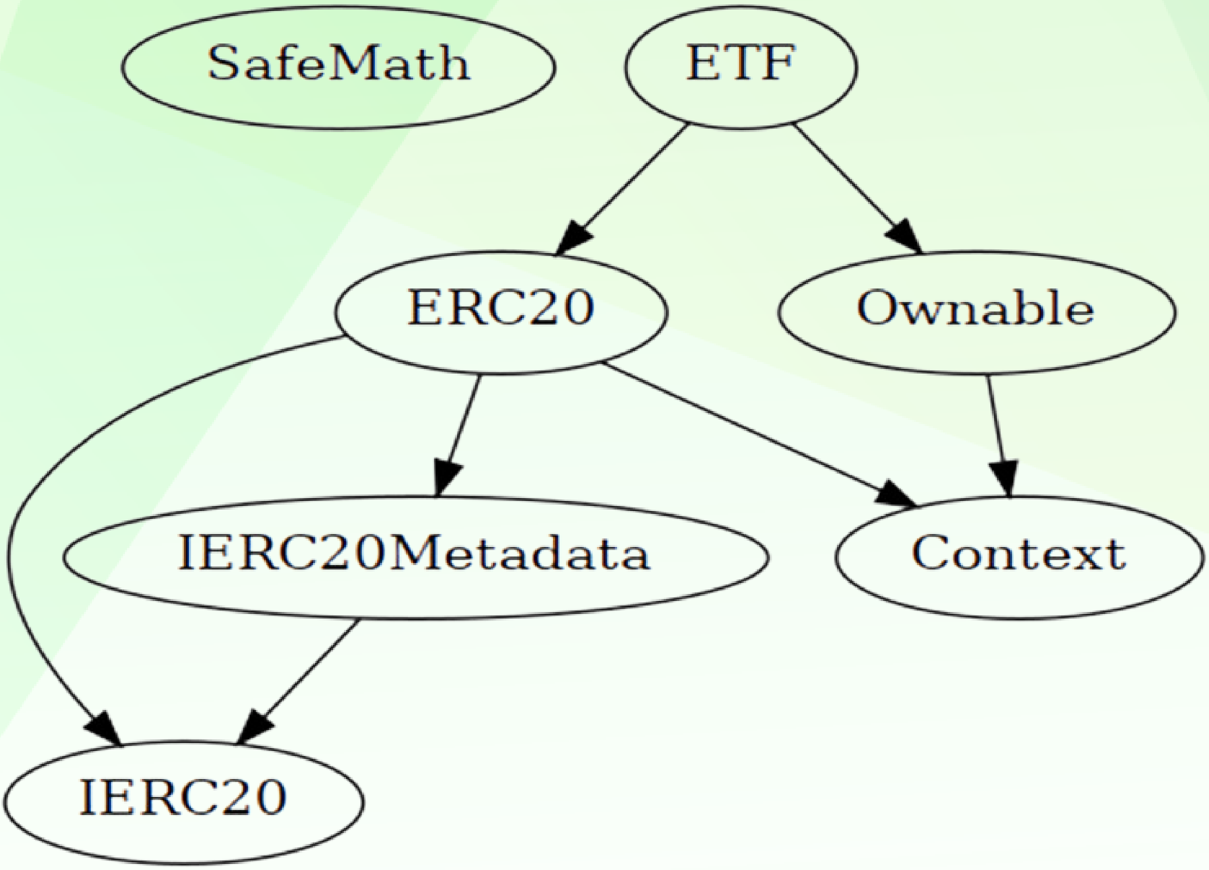
CLASSIFICATION OF RISK

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	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set buy/sell/transfer fees

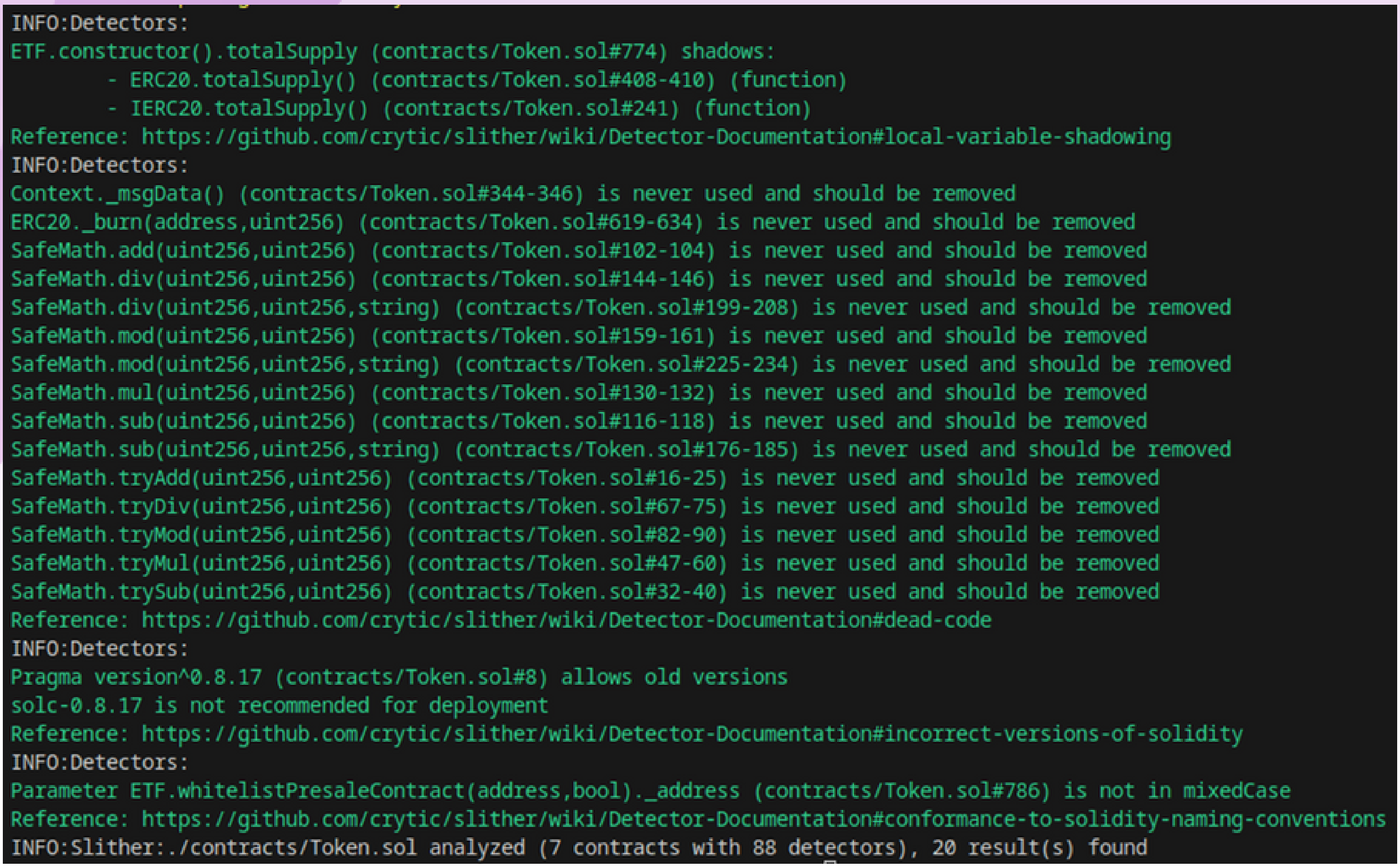
- Owner is not able to blacklist an arbitrary wallet

- Owner is not able to disable trades

- Owner is not able to mint new tokens

- Owner must enable trades manually

STATIC ANALYSIS



```
INFO:Detectors:
ETF.constructor().totalSupply (contracts/Token.sol#774) shadows:
        - ERC20.totalSupply() (contracts/Token.sol#408-410) (function)
        - IERC20.totalSupply() (contracts/Token.sol#241) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Context._msgData() (contracts/Token.sol#344-346) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#619-634) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/Token.sol#102-104) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/Token.sol#144-146) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/Token.sol#199-208) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/Token.sol#159-161) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/Token.sol#225-234) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/Token.sol#130-132) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/Token.sol#116-118) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/Token.sol#176-185) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (contracts/Token.sol#16-25) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (contracts/Token.sol#67-75) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (contracts/Token.sol#82-90) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (contracts/Token.sol#47-60) is never used and should be removed
SafeMath.trySub(uint256,uint256) (contracts/Token.sol#32-40) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/Token.sol#8) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Parameter ETF.whitelistPresaleContract(address,bool)._address (contracts/Token.sol#786) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Slither:./contracts/Token.sol analyzed (7 contracts with 88 detectors), 20 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

CONTRACT ASSESMENT

```
| Contract|      Type     |Bases    |            |            |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
|    └| **Function Name** |**Visibility** | **Mutability** |**Modifiers** |
||||||
| **SafeMath** | Library | |||
| └ | tryAdd | Internal 🔒 | | |
| └ | trySub | Internal 🔒 | | |
| └ | tryMul | Internal 🔒 | | |
| └ | tryDiv | Internal 🔒 | | |
| └ | tryMod | Internal 🔒 | | |
| └ | add | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | mul | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | mod | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | mod | Internal 🔒 | | |
||||||
| **IERC20** | Interface | |||
| └ | totalSupply | External ❗ | |NO ❗ |
| └ | balanceOf | External ❗ | |NO ❗ |
| └ | transfer | External ❗ | 🛑 |NO ❗ |
| └ | allowance | External ❗ | |NO ❗ |
| └ | approve | External ❗ | 🛑 |NO ❗ |
| └ | transferFrom | External ❗ | 🛑|NO ❗ |
||||||
| **IERC20Metadata** | Interface | IERC20 |||
| └ | name | External ❗ | |NO ❗ |
| └ | symbol | External ❗ | |NO ❗ |
| └ | decimals | External ❗ | |NO ❗ |
||||||
```

CONTRACT ASSESMENT

```
| **Context** | Implementation | |||
| └ | _msgSender | Internal 🔒 | | |
| └ | _msgData | Internal 🔒 | | |
||||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata |||
| └ | <Constructor> | Public ❗ | 🛑|NO ❗ |
| └ | name | Public ❗ | |NO ❗ |
| └ | symbol | Public ❗ | |NO ❗ |
| └ | decimals | Public ❗ | |NO ❗ |
| └ | totalSupply | Public ❗ | |NO ❗ |
| └ | balanceOf | Public ❗ | |NO ❗ |
| └ | transfer | Public ❗ | 🛑 |NO ❗ |
| └ | allowance | Public ❗ | |NO ❗ |
| └ | approve | Public ❗ | 🛑 |NO ❗ |
| └ | transferFrom | Public ❗ | 🛑|NO ❗ |
| └ | increaseAllowance | Public ❗ | 🛑|NO ❗ |
| └ | decreaseAllowance | Public ❗ | 🛑|NO ❗ |
| └ | _transfer | Internal 🔒 | 🛑| |
| └ | _mint | Internal 🔒 | 🛑 | |
| └ | _burn | Internal 🔒 | 🛑 | |
| └ | _approve | Internal 🔒 | 🛑| |
| └ | _beforeTokenTransfer | Internal 🔒 | 🛑 | |
| └ | _afterTokenTransfer | Internal 🔒 | 🛑 | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑|NO ❗ |
| └ | owner | Public ❗ | |NO ❗ |
| └ | renounceOwnership | Public ❗ | 🛑| onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑| onlyOwner |
| └ | _transferOwnership | Internal 🔒 | 🛑 | |
||||||
| **ETF** | Implementation | ERC20, Ownable |||
| └ | <Constructor> | Public ❗ | 🛑| ERC20 |
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |
| └ | startTrading | External ❗ | 🛑 | onlyOwner |
| └ | whitelistPresaleContract | External ❗ | 🛑 | onlyOwner |
| └ | _transfer | Internal 🔒 | 🛑| |
```

CONTRACT ASSESMENT

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

FUNCTIONAL TESTING

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0x872c2179c2f46c9e215eecfdacbb8efe3638c6ae861229cc259352ce1b4864d6

2- Buying (0% tax) (passed):

https://testnet.bscscan.com/tx/0xbfcf3173e6ad6109659affc12cb6838eedb25cf35fa6a07682afe00910cee2a9

3- Selling (0% tax) (passed):

https://testnet.bscscan.com/tx/0x543a5829766ab0b5e5230b7f72534f14176e05939032ce182982c0c2777fad5e

4- Transferring (0% tax) (passed):

https://testnet.bscscan.com/tx/0xf5b78d98f19b4f25701044e6836db3a52aa8dcca2132928c3cb7460de2344e10

MANUAL TESTING

Centralization – Enabling Trades
Severity: High
function: startTrading
Status: Open

Overview:

The startTrading 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 startTrading() 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

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