Project: solidify

0x88490ce333e3f3aa384e5183836142490600da8c

12/03/2024

AUDIT REPORT

SAFETY SCORE: 85

1 - Arbitrary Jump/Storage Write

Result: Pass

2 - Centralization of Control

Result: Medium

Details: The contract has an `onlyOwner` modifier which centralizes control to the owner for certain functions such as `initialize`, `launch`,

`toggleLimits`, and `setTax`. This centralization can be a risk if the owner's account is compromised or if the owner acts maliciously.

3 - Compiler Issues

Result: Pass

4 - Delegate Call to Untrusted Contract

Result: Pass

5 - Dependence on Predictable Variables

Result: Pass

6 - Ether/Token Theft

Result: Pass

7 - Flash Loans

Result: Pass

8 - Front Running

Result: Medium

Details: The contract is vulnerable to front running because it does not implement any mechanism to prevent it, such as using a

commit-reveal scheme or similar. This can be exploited by miners or bots that can see pending transactions and execute their own

transactions first with higher gas fees.

9 - Improper Events

Result: Pass

10 - Improper Authorization Scheme

Result: Pass

11 - Integer Over/Underflow

Result: Pass

Details: The contract uses SafeMath library for all arithmetic operations, which protects against integer overflow and underflow.

12 - Logical Issues

Result: Medium

Details: The `initialize` function can be called by the owner even after trading has been opened, which could potentially be used to

manipulate the contract's state in an undesirable way. Additionally, the `launch` function can be called multiple times as long as

`tradingOpen` is false, which could lead to unexpected behavior.

13 - Oracle Issues

Result: Pass

14 - Outdated Compiler Version

Result: Pass

15 - Race Conditions

Result: Pass

16 - Reentrancy

Result: Pass

17 - Signature Issues

Result: Pass

18 - Sybil Attack

Result: Pass

19 - Unbounded Loops

Result: Pass

20 - Unused Code

Result: Low

Details: The contract contains constants that are never used, such as `_preventSwapBefore`, which could lead to confusion and should be

removed if not needed.

Code:

```
uint256 private constant _preventSwapBefore=25;
```

Correction:

```
// Removed unused constant

// uint256 private constant _preventSwapBefore=25;
```

To address the issues found:

Centralization of Control:

- Consider implementing a multi-signature scheme or a decentralized governance model for critical functions.

Front Running:

- Implement measures to prevent front running, such as using a commit-reveal scheme.

Logical Issues:

- Ensure that functions like `initialize` and `launch` cannot be called inappropriately.

Code:

```
function initialize () external onlyOwner {

require(!tradingOpen,"init already called");

// Rest of the code

}

function launch() external onlyOwner {

require(!tradingOpen,"trading already open");

// Rest of the code

}
```

Correction:

```
function initialize () external onlyOwner {

require(!tradingOpen && !initialized,"init already called");

initialized = true;
```

```
// Rest of the code

}


function launch() external onlyOwner {

require(!tradingOpen && initialized,"trading already open or
not initialized");

tradingOpen = true;

// Rest of the code

}
```

- Add a state variable `initialized` to ensure `initialize` can only be called once.

Unused Code:

- Remove or comment out unused constants and variables to clean up the contract and reduce confusion.