Scope

The commit reviewed was 50048fbb437e5b348c1c7d0968644fec2c99db50. The review covered the entire repository at this specific commit but focused on the contracts directory. The review is a code review to identify potential vulnerabilities in the code. The reviewers did not investigate security practices or operational security and assumed that privileged accounts could be trusted. The reviewers did not evaluate the security of the code relative to a standard or specification. The review may not have identified all potential attack vectors or areas of vulnerability.The reviewer does not intend to perform re-review after fixes.

Findings

Explanation Findings are broken down into sections by their respective impact:

- Critical, High, Medium, Low impact
    - These are findings that range from attacks that may cause loss of funds, impact control/ownership of the contracts, or cause any unintended consequences/actions that are outside the scope of the requirements,
- Gas savings
    - Findings that can improve the gas efficiency of the contracts
- Informational
    - Findings including recommendations and best practices

High findings

None found

Medium findings

1)

unused-return

Impact: Medium Confidence: Medium

ID-3 TrueYield.closePosition(uint256) ignores return value by IERC20(aWethAddress).approve(address(iWethGateway),type()(uint256).max)

../contracts/TrueYield.sol#L134-L156

Recommendation:
Replace code:

```
IERC20(aWethAddress).approve(address(iWethGateway), type(uint256).max);
```

With:

```
(bool success, ) =
IERC20(aWethAddress).approve(address(iWethGateway), type(uint256).max);
require(success, "Transaction failed");
```

Low findings

1)

variable-scope

Impact: Low Confidence: High

> ID-4 Variable 'TrueYield.closePosition(uint256).success' in TrueYield.closePosition(uint256) potentially used before declaration: (success) = address(msg.sender).call{value: positions[positionId].weiStaked}()

../contracts/TrueYield.sol#L149

Recommendation:
Recommendation
Move all variable declarations prior to any usage of the variable, and ensure that reaching a variable declaration does not depend on some conditional if it is used unconditionally.

2)

boolean-equal

../contracts/TrueYield.sol#L134-L156

Recommendation:

```
require(positions[positionId].open == true, "Position is closed");
```

Replace with:
```
require(positions[positionId].open, "Position is closed");
```

3)
Unused variable
```
Position private position;
```
../contracts/TrueYield.sol#L47

Recommendation:
Either delete this line or make a explicit comment why it should stay like this.

Compilation warnings

Although not a direct security vulnerability, compilation warnings indicate missing opportunities to improve code clarity and readability which make development more error-prone. I would mark them as Medium findings:

```
warning[5667]: Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
   --> contracts/TrueYield.sol:108:50:
    |
108 |     function calculateInterest(uint basisPoints, uint numDays, uint weiAmount) public pure returns (uint) {
    |                                                  ^^^^^^^^^^^^
```

```
warning[5667]: Warning: Unused function parameter. Remove or comment out the variable name to silence this warning.
  --> contracts/mocks/MockTrueYield.sol:90:50:
   |
90 |     function calculateInterest(uint basisPoints, uint numDays, uint weiAmount) public pure returns (uint) {
   |                                                  ^^^^^^^^^^^^
```