DYP-VAULT SMART CONTRACT AUDIT

April 2021

BLOCKCHAIN CONSILIUM

Contents

Disclaimer

THE AUDIT MAKES NO STATEMENTS OR WARRANTIES ABOUT UTILITY OF THE CODE, SAFETY OF THE CODE, SUITABILITY OF THE BUSINESS MODEL, REGULATORY REGIME FOR THE BUSINESS MODEL, OR ANY OTHER STATEMENTS ABOUT FITNESS OF THE CONTRACTS TO PURPOSE, OR THEIR BUG FREE STATUS. THE AUDIT DOCUMENTATION IS FOR DISCUSSION PURPOSES ONLY.

THE CONTENT OF THIS AUDIT REPORT IS PROVIDED "AS IS", WITHOUT REPRESENTATIONS AND WARRANTIES OF ANY KIND, AND BLOCKCHAIN CONSILIUM DISCLAIMS ANY LIABILITY FOR DAMAGE ARISING OUT OF, OR IN CONNECTION WITH, THIS AUDIT REPORT. COPYRIGHT OF THIS REPORT REMAINS WITH BLOCKCHAIN CONSILIUM.

Purpose of the report

The Audits and the analysis described therein are created solely for Clients and published with their consent. The scope of our review is limited to a review of Solidity code and only the Solidity code we note as being within the scope of our review within this report. The Solidity language itself remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the Solidity programming language that could present security risks. Cryptographic tokens and smart contracts are emergent technologies and carry with them high levels of technical risk and uncertainty.

The Audits are not an endorsement or indictment of any particular project or team, and the Audits do not guarantee the security of any particular project. This Report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any Third-Party in any respect, including regarding the bugfree nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. No third party should rely on the Audits in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. This Report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. There is no owed duty to any Third-Party by virtue of publishing these Audits.

Copyright 2021, Blockchain Consilium.

Introduction

We first thank dyp.finance for giving us the opportunity to audit their smart contract. This document outlines our methodology, audit details, and results.

dyp.finance asked us to review their DYP Earn Vault smart contract (GitHub Commit Hash: `8bf1eea9f88be9d26bfc93311bdec0d503f1d405`). Blockchain Consilium reviewed the system from a technical perspective looking for bugs, issues and vulnerabilities in their code base. The Audit is valid for above mentioned GitHub Commit Hash(es) only. The audit is not valid for any other versions of the smart contract. Read more below.

Audit Summary

This code is clean, thoughtfully written and in general well architected. The code conforms closely to the documentation and specification.

Overall, the code is clear on what it is supposed to do for each function. The visibility and state mutability of all the functions are clearly specified, and there are no confusions.

https://github.com/dypfinance/dyp-earn-vault/tree/8bf1eea9f88be9d26bfc93311bdec0d503f1d405

Audit Result	**PASSED AS PER SPECS, 1 INFORMATIONAL OBSERVATION**
High Severity Issues	None
Moderate Severity Issues	None
Low Severity Issues	None
Informational Observations	1

Overview

The project has two Solidity files, **vault-weth.sol** and **vault.sol**, one for WETH vault and one for Token Vaults, both are similar in implementation except the WETH vault uses CEth Compound interface and other uses CErc20 Compound Interface, containing about 1,390 and 1,375 lines of code respectively. We manually reviewed each line of code in the smart contract.

Methodology

Blockchain Consilium manually reviewed the smart contract line-by-line, keeping in mind industry best practices and known attacks, looking for any potential issues and vulnerabilities, and areas where improvements are possible.

We also used automated tools like slither / surya for analysis and reviewing the smart contract. The raw output of these tools is included in the Appendix. These tools often give false-positives, and any issues reported by them but not included in the issue list can be considered not valid.

Classification / Issue Types Definition:

1. **High Severity:** which presents a significant security vulnerability or failure of the contract across a range of scenarios, or which may result in loss of funds.
2. **Moderate Severity:** which affects the desired outcome of the contract execution or introduces a weakness that can be exploited. It may not result in loss of funds but breaks the functionality or produces unexpected behaviour.
3. **Low Severity:** which does not have a material impact on the contract execution and is likely to be subjective.

The smart contract is considered to pass the audit, as of the audit date, if no high severity or moderate severity issues are found.

Smart Contract Overview

The DYP Earn Vault Smart Contracts let users deposit a particular token, lock for a certain length of time, and claim rewards available from Compound, deposit and withdraw fees, and some ether fees as well on transactions. Additionally DYP Admin is expected to supply DYP to the contracts to cover the DYP rewards given out at constant rate over time.

Fees from transactions and deposit / withdraw fee is distributed pro-rata to the vault users after using a set % of the fee to buy back and burn DYP from Uniswap at spot price. DYP rewards, provided by DYP admin, are distributed constantly to users over time.

In case sufficient DYP rewards are not available and any users pending balance exceeds the available DYP balance, they cannot claim their DYP rewards. Thus it is recommended to frequently claim rewards, and for admins to monitor and supply the DYP rewards over time in sufficient amounts.

There's an emergencyWithdraw function which allows users to withdraw their deposit without interacting with Uniswap, in this case the fees will remain undistributed in the contract.

After the contract ends, mostly one year from deployment, the admin can withdraw any remaining tokens from the smart contract.

It must be noted that while claiming DYP rewards, Uniswap spot price is used, usually such interactions are vulnerable to sandwiching exploits and frontrunning or backrunning and Uniswap price manipulation. Although the `noContractsAllowed` modifier mitigates this by limiting flash loan based exploits to some extent, and the slippage control mitigates potential frontrunning or backrunning, however the sandwiching issue is not completely eliminated. This issue is largely inherent to the current blockchain infrastructure and there's still need to search for better and effective defence.

Attacks & Issues considered while auditing

In order to check for the security of the contract, we reviewed each line of code in the smart contract considering several known Smart Contract Attacks & known issues.

- ### Overflows and underflows

  An overflow happens when the limit of the type variable uint256 , 2 ** 256, is exceeded. What happens is that the value resets to zero instead of incrementing more.

  For instance, if we want to assign a value to a uint bigger than 2 ** 256 it will simple go to 0—this is dangerous.

  On the other hand, an underflow happens when you try to subtract 0 minus a number bigger than 0.For example, if you subtract 0 - 1 the result will be = 2 ** 256 instead of -1.

  This is quite dangerous. This contract **DOES** check for overflows and underflows using **OpenZeppelin's** *SafeMath* for overflow and underflow protection.

- ### Reentrancy Attack

  One of the major dangers of calling external contracts is that they can take over the control flow, and make changes to your data that the calling function wasn't expecting. This class of bug can take many forms, and both of the major bugs that led to the DAO's collapse were bugs of this sort.

This smart contract uses OpenZeppelin's ReentrancyGuard to protect against this attack.

- **Replay attack**

  The replay attack consists of making a transaction on one blockchain like the original Ethereum's blockchain and then repeating it on another blockchain like the Ethereum's classic blockchain. The ether is transferred like a normal transaction from a blockchain to another. Though it's no longer a problem because since the version 1.5.3 of *Geth* and 1.4.4 of *Parity* both implement the attack protection EIP 155 by Vitalik Buterin.

  So the people that will use the contract depend on their own ability to be updated with those programs to keep themselves secure.

- **Short address attack**

  This attack affects ERC20 tokens, was discovered by the Golem team and consists of the following:

  A user creates an Ethereum wallet with a trailing 0, which is not hard because it's only a digit. For instance: `0xiofa8d97756as7df5sd8f75g8675ds8gsdg0`
  Then he buys tokens by removing the last zero:
  Buy 1000 tokens from account `0xiofa8d97756as7df5sd8f75g8675ds8gsdg`. If the contract has enough amount of tokens and the buy function doesn't check the length of the address of the sender, the Ethereum's virtual machine will just add zeroes to the transaction until the address is complete.

  The virtual machine will return 256000 for each 1000 tokens bought. This is abug of the virtual machine.

  Here is a **fix for short address attacks**

```
modifier onlyPayloadSize(uint size) {
    assert(msg.data.length >= size + 4);
    _;
}
function transfer(address _to, uint256 _value) onlyPayloadSize(2 * 32) {
    // do stuff
}
```

Whether or not it is appropriate for token contracts to mitigate the short-address attack is a contentious issue among smart-contract developers. Many, including those behind the OpenZeppelin project, have explicitly chosen not to *do so. Blockchain Consilium doesn't consider short address attack an issue of the smart contract at the token smart contract level.*

This contract is not an ERC20 Token and thus is not found vulnerable to ERC20 Short Address Attacks.

You can read more about the attack here: ERC20 Short Address Attacks.

- **Approval Double-spend**

  ERC20 Standard allows users to approve other users to manage their tokens, or spend tokens from their account till a certain amount, by setting the user's allowance with the standard `approve` function, then the allowed user may use `transferFrom` to spend the allowed tokens.

  Hypothetically, given a situation where Alice approves Bob to spend 100 Tokens from her account, and if Alice needs to adjust the allowance to allow Bob to spend 20 more tokens, normally – she'd check Bob's allowance (100 currently) and start a new `approve` transaction allowing Bob to spend a total of 120 Tokens instead of 100 Tokens.

  Now, if Bob is monitoring the Transaction pool, and as soon as he observes new transaction from Alice approving more amount, he may send a `transferFrom` transaction spending 100 Tokens from Alice's account with higher gas price and do all the required effort to get his spend transaction mined before Alice's new approve transaction.

  Now Bob has already spent 100 Tokens, and given Alice's approve transaction is mined, Bob's allowance is set to 120 Tokens, this would allow Bob to spend a total of 100 + 120 = 220 Tokens from Alice's account instead of the allowed 120 Tokens. This exploit situation is known as Approval Double-Spend Attack.

  A potential solution to minimize these instances would be to set the non-zero allowance to 0 before setting it to any other amount.

  It's possible for `approve` to enforce this behaviour without interface changes in the ERC20 specification:

```
if ((_value != 0) && (approved[msg.sender][_spender] != 0)) return false;
```

  However, this is just an attempt to modify user behaviour. If the user does attempt to change from one non-zero value to another, the double spend might still happen, since the attacker may set the value to zero by already spending all the previously allowed value before the user's new approval transaction.

Copyright 2021, Blockchain Consilium.

If desired, a non-standard function can be added to minimize hassle for users:

```
function increaseAllowance (address _spender, uint256 _addedValue)
returns (bool success) {
  uint oldValue = approved[msg.sender][_spender];
  approved[msg.sender][_spender] = safeAdd(oldValue, _addedValue);
  return true;
}
```

Even if this function is added, it’s important to keep the original for compatibility with the ERC20 specification.

Likely impact of this bug is low for most situations. This contract is not an ERC20 token, and thus it is not found vulnerable to Approval Doublespend attacks.

For more, see this discussion on GitHub:
https://github.com/ethereum/EIPs/issues/20#issuecomment263524729

- **Sybil attacks**
  In a Sybil attack, the attacker subverts the reputation system of a network service by creating a large number of pseudonymous identities and uses them to gain a disproportionately large influence.

  Normally in ERC20 & DeFi smart contracts, sybil attacks are related to voting power amplification where a user may use their vote more than once in case of governance tokens. This smart contract does not implement governance or voting and thus is not found vulnerable to this specific type of sybil attacks in smart contracts & tokens.

Copyright 2021, Blockchain Consilium.

Issues Found

High Severity Issues

No high severity issues were found in the smart contract.

Moderate Severity Issues

No moderate severity issues were found in the smart contract.

Low Severity Issues

No low severity issues were found in the smart contract.

Informational Observations

It must be noted that while claiming DYP rewards, Uniswap spot price is used, usually such interactions are vulnerable to sandwiching exploits and frontrunning or backrunning and Uniswap price manipulation. Although the `noContractsAllowed` modifier mitigates this by limiting flash loan based exploits to some extent, and the slippage control mitigates potential frontrunning or backrunning, however the sandwiching issue is not completely eliminated.

This issue is largely inherent to the current blockchain infrastructure and there's still need to search for better and effective defence.

It is recommended to supply DYP rewards over time in smaller quantities to the smart contract rather than adding all at once, for example, DYP rewards can be supplied to cover weekly or monthly demand instead of covering yearly demands, to reduce the risk of sandwiching or exploits a bit more.

It is also recommended to monitor the system for any unexpected activities and act accordingly.

If the contract has insufficient DYP balance to cover the user's pending rewards, the users will be unable to claim their DYP rewards. The users must be informed of an appropriate time to unstake their rewards, since this contract is meant to be used for limited time.

Copyright 2021, Blockchain Consilium.

Appendix

Smart Contract Summary

vault.sol

- Contract SafeMath (Most derived contract)

    - From SafeMath
        - add(uint256,uint256) (internal)
        - div(uint256,uint256) (internal)
        - div(uint256,uint256,string) (internal)
        - mod(uint256,uint256) (internal)
        - mod(uint256,uint256,string) (internal)
        - mul(uint256,uint256) (internal)
        - sub(uint256,uint256) (internal)
        - sub(uint256,uint256,string) (internal)

- Contract Address (Most derived contract)

    - From Address
        - _functionCallWithValue(address,bytes,uint256,string) (private)
        - functionCall(address,bytes) (internal)
        - functionCall(address,bytes,string) (internal)
        - functionCallWithValue(address,bytes,uint256) (internal)
        - functionCallWithValue(address,bytes,uint256,string) (internal)
        - isContract(address) (internal)
        - sendValue(address,uint256) (internal)

- Contract EnumerableSet (Most derived contract)

    - From EnumerableSet
        - _add(EnumerableSet.Set,bytes32) (private)
        - _at(EnumerableSet.Set,uint256) (private)
        - _contains(EnumerableSet.Set,bytes32) (private)
        - _length(EnumerableSet.Set) (private)
        - _remove(EnumerableSet.Set,bytes32) (private)
        - add(EnumerableSet.AddressSet,address) (internal)
        - add(EnumerableSet.UintSet,uint256) (internal)
        - at(EnumerableSet.AddressSet,uint256) (internal)
        - at(EnumerableSet.UintSet,uint256) (internal)
        - contains(EnumerableSet.AddressSet,address) (internal)

Copyright 2021, Blockchain Consilium.

- contains(EnumerableSet.UintSet,uint256) (internal)
- length(EnumerableSet.AddressSet) (internal)
- length(EnumerableSet.UintSet) (internal)
- remove(EnumerableSet.AddressSet,address) (internal)
- remove(EnumerableSet.UintSet,uint256) (internal)

- Contract SafeERC20 (Most derived contract)
    - From SafeERC20
        - _callOptionalReturn(IERC20,bytes) (private)
        - safeApprove(IERC20,address,uint256) (internal)
        - safeDecreaseAllowance(IERC20,address,uint256) (internal)
        - safeIncreaseAllowance(IERC20,address,uint256) (internal)
        - safeTransfer(IERC20,address,uint256) (internal)
        - safeTransferFrom(IERC20,address,address,uint256) (internal)
- Contract IERC20 (Most derived contract)
    - From IERC20
        - allowance(address,address) (external)
        - approve(address,uint256) (external)
        - balanceOf(address) (external)
        - totalSupply() (external)
        - transfer(address,uint256) (external)
        - transferFrom(address,address,uint256) (external)
- Contract ReentrancyGuard
    - From ReentrancyGuard
        - constructor() (internal)
- Contract Ownable
    - From Ownable
        - constructor() (internal)
        - owner() (public)
        - renounceOwnership() (public)
        - transferOwnership(address) (public)
- Contract CErc20 (Most derived contract)
    - From CErc20
        - exchangeRateCurrent() (external)
        - exchangeRateStored() (external)
        - mint(uint256) (external)

Copyright 2021, Blockchain Consilium.

        - redeem(uint256) (external)
        - redeemUnderlying(uint256) (external)
        - supplyRatePerBlock() (external)

- Contract IUniswapV2Router (Most derived contract)
    - From IUniswapV2Router
        - WETH() (external)
        - getAmountsOut(uint256,address[]) (external)
        - swapExactETHForTokens(uint256,address[],address,uint256) (external)
        - swapExactTokensForTokens(uint256,uint256,address[],address,uint256) (external)

- Contract Vault (Most derived contract)
    - From Ownable
        - owner() (public)
        - renounceOwnership() (public)
        - transferOwnership(address) (public)
    - From Vault
        - _claimCompoundDivs() (private)
        - _claimEthDivs() (private)
        - _claimPlatformTokenDivs(uint256) (private)
        - _claimTokenDivs() (private)
        - addPlatformTokenBalance(uint256) (external)
        - claim(uint256) (external)
        - claimAnyToken(address,uint256) (external)
        - claimCompoundDivs() (external)
        - claimEthDivs() (external)
        - claimExtraTokens(address) (external)
        - claimPlatformTokenDivs(uint256) (external)
        - claimTokenDivs() (external)
        - constructor() (public)
        - decreaseTokenBalance(address,uint256) (private)
        - deposit(uint256,uint256,uint256) (external)
        - distributeEthDivs(uint256) (private)
        - distributeTokenDivs(uint256) (private)
        - emergencyWithdraw(uint256) (external)
        - ethDivsOwing(address) (public)
        - getConvertedBalance(uint256) (public)
        - getDepositorsList(uint256,uint256) (public)
        - getEstimatedCompoundDivsOwing(address) (public)

Copyright 2021, Blockchain Consilium.

        - getExchangeRateCurrent() (public)
        - getExchangeRateStored() (public)
        - getNumberOfHolders() (public)
        - handleEthFee(uint256,uint256,uint256) (private)
        - handleFee(uint256,uint256,uint256) (private)
        - increaseTokenBalance(address,uint256) (private)
        - platformTokenDivsOwing(address) (public)
        - receive() (external)
        - tokenDivsOwing(address) (public)
        - updateAccount(address) (private)
        - withdraw(uint256,uint256,uint256,uint256) (external)

vault-weth.sol

- Contract SafeMath (Most derived contract)
    - From SafeMath
        - add(uint256,uint256) (internal)
        - div(uint256,uint256) (internal)
        - div(uint256,uint256,string) (internal)
        - mod(uint256,uint256) (internal)
        - mod(uint256,uint256,string) (internal)
        - mul(uint256,uint256) (internal)
        - sub(uint256,uint256) (internal)
        - sub(uint256,uint256,string) (internal)
- Contract Address (Most derived contract)
    - From Address
        - _functionCallWithValue(address,bytes,uint256,string) (private)
        - functionCall(address,bytes) (internal)
        - functionCall(address,bytes,string) (internal)
        - functionCallWithValue(address,bytes,uint256) (internal)
        - functionCallWithValue(address,bytes,uint256,string) (internal)
        - isContract(address) (internal)
        - sendValue(address,uint256) (internal)
- Contract EnumerableSet (Most derived contract)
    - From EnumerableSet
        - _add(EnumerableSet.Set,bytes32) (private)

        - _at(EnumerableSet.Set,uint256) (private)
        - _contains(EnumerableSet.Set,bytes32) (private)
        - _length(EnumerableSet.Set) (private)
        - _remove(EnumerableSet.Set,bytes32) (private)
        - add(EnumerableSet.AddressSet,address) (internal)
        - add(EnumerableSet.UintSet,uint256) (internal)
        - at(EnumerableSet.AddressSet,uint256) (internal)
        - at(EnumerableSet.UintSet,uint256) (internal)
        - contains(EnumerableSet.AddressSet,address) (internal)
        - contains(EnumerableSet.UintSet,uint256) (internal)
        - length(EnumerableSet.AddressSet) (internal)
        - length(EnumerableSet.UintSet) (internal)
        - remove(EnumerableSet.AddressSet,address) (internal)
        - remove(EnumerableSet.UintSet,uint256) (internal)

- Contract SafeERC20 (Most derived contract)
    - From SafeERC20
        - _callOptionalReturn(IERC20,bytes) (private)
        - safeApprove(IERC20,address,uint256) (internal)
        - safeDecreaseAllowance(IERC20,address,uint256) (internal)
        - safeIncreaseAllowance(IERC20,address,uint256) (internal)
        - safeTransfer(IERC20,address,uint256) (internal)
        - safeTransferFrom(IERC20,address,address,uint256) (internal)
- Contract IERC20 (Most derived contract)
    - From IERC20
        - allowance(address,address) (external)
        - approve(address,uint256) (external)
        - balanceOf(address) (external)
        - totalSupply() (external)
        - transfer(address,uint256) (external)
        - transferFrom(address,address,uint256) (external)
- Contract ReentrancyGuard
    - From ReentrancyGuard
        - constructor() (internal)
- Contract Ownable
    - From Ownable
        - constructor() (internal)

    - owner() (public)
    - renounceOwnership() (public)
    - transferOwnership(address) (public)

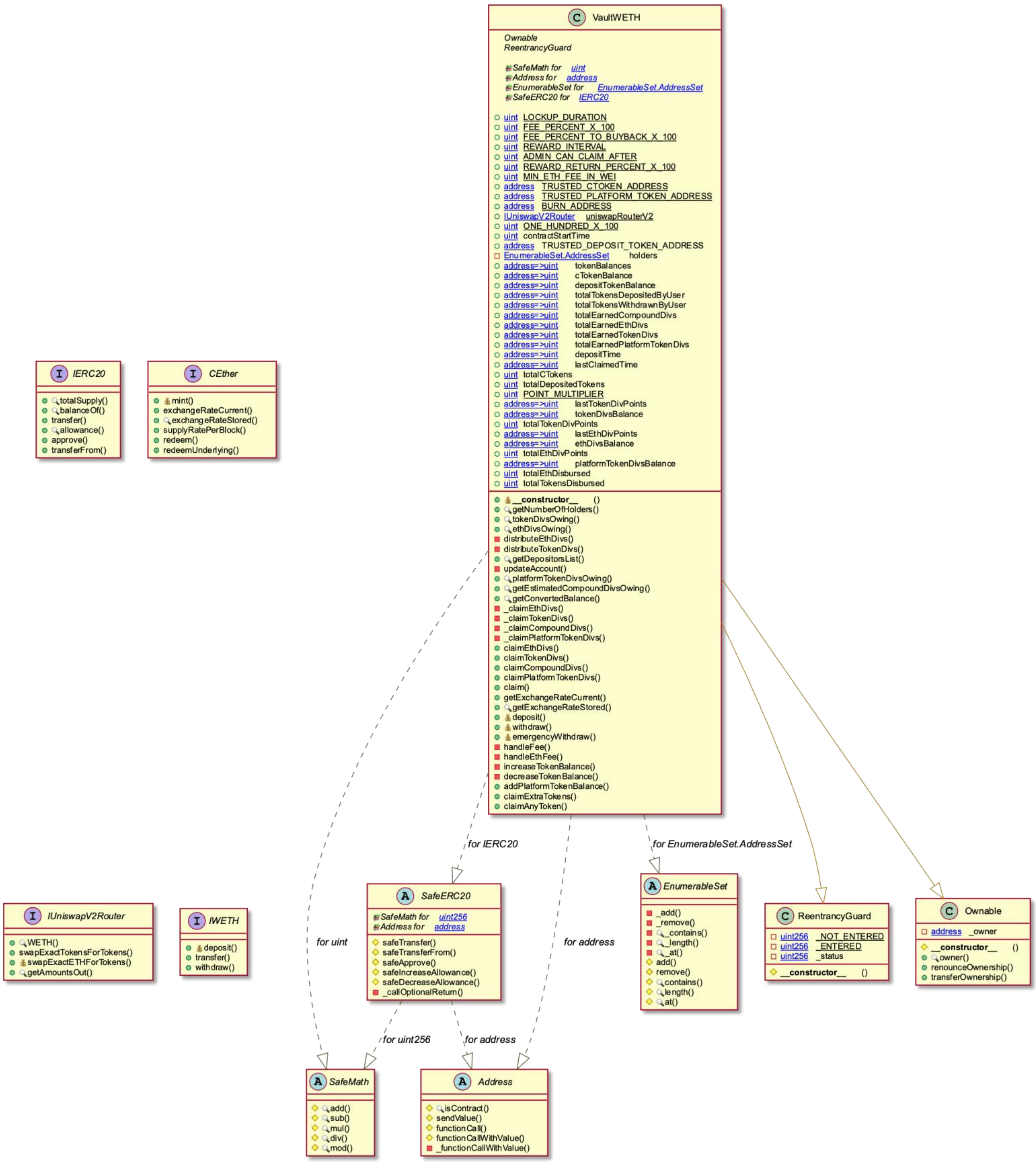
- Contract CEther (Most derived contract)
  - From CEther
    - exchangeRateCurrent() (external)
    - exchangeRateStored() (external)
    - mint() (external)
    - redeem(uint256) (external)
    - redeemUnderlying(uint256) (external)
    - supplyRatePerBlock() (external)
- Contract IUniswapV2Router (Most derived contract)
  - From IUniswapV2Router
    - WETH() (external)
    - getAmountsOut(uint256,address[]) (external)
    - swapExactETHForTokens(uint256,address[],address,uint256) (external)
    - swapExactTokensForTokens(uint256,uint256,address[],address,uint256) (external)
- Contract IWETH (Most derived contract)
  - From IWETH
    - deposit() (external)
    - transfer(address,uint256) (external)
    - withdraw(uint256) (external)
- Contract VaultWETH (Most derived contract)
  - From Ownable
    - owner() (public)
    - renounceOwnership() (public)
    - transferOwnership(address) (public)
  - From VaultWETH
    - _claimCompoundDivs() (private)
    - _claimEthDivs() (private)
    - _claimPlatformTokenDivs(uint256) (private)
    - _claimTokenDivs() (private)
    - addPlatformTokenBalance(uint256) (external)
    - claim(uint256) (external)
    - claimAnyToken(address,uint256) (external)

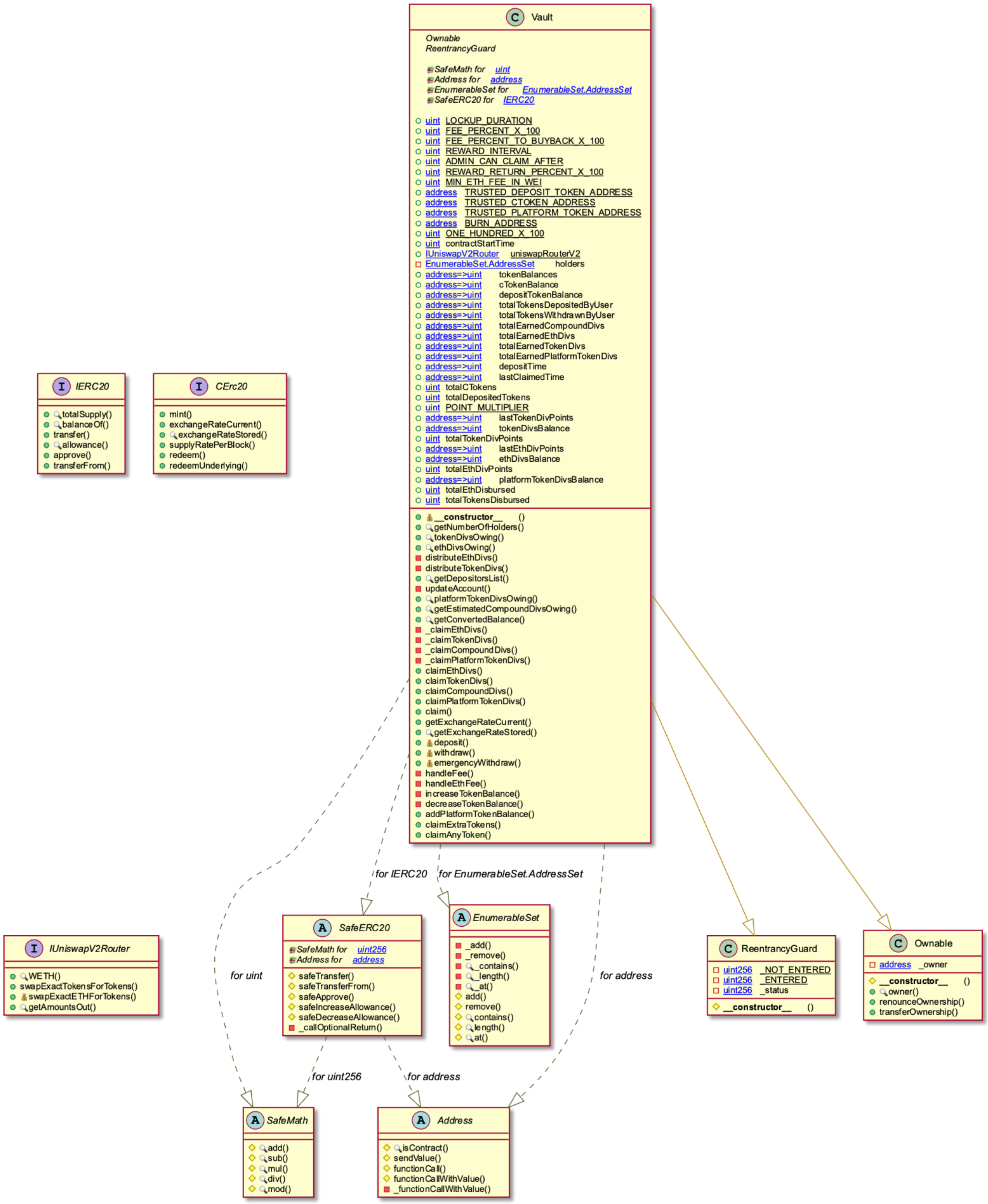
Copyright 2021, Blockchain Consilium.

- claimCompoundDivs() (external)
- claimEthDivs() (external)
- claimExtraTokens(address) (external)
- claimPlatformTokenDivs(uint256) (external)
- claimTokenDivs() (external)
- constructor() (public)
- decreaseTokenBalance(address,uint256) (private)
- deposit(uint256,uint256,uint256) (external)
- distributeEthDivs(uint256) (private)
- distributeTokenDivs(uint256) (private)
- emergencyWithdraw(uint256) (external)
- ethDivsOwing(address) (public)
- getConvertedBalance(uint256) (public)
- getDepositorsList(uint256,uint256) (public)
- getEstimatedCompoundDivsOwing(address) (public)
- getExchangeRateCurrent() (public)
- getExchangeRateStored() (public)
- getNumberOfHolders() (public)
- handleEthFee(uint256,uint256,uint256) (private)
- handleFee(uint256,uint256,uint256) (private)
- increaseTokenBalance(address,uint256) (private)
- platformTokenDivsOwing(address) (public)
- receive() (external)
- tokenDivsOwing(address) (public)
- updateAccount(address) (private)
- withdraw(uint256,uint256,uint256,uint256) (external)

Copyright 2021, Blockchain Consilium.

Inheritance Graph and UML Diagram




Vault
Ownable
ReentrancyGuard
SafeMath for uint
Address for address
EnumerableSet for EnumerableSet.AddressSet
SafeERC20 for IERC20
uint LOCKUP_DURATION
uint FEE_PERCENT_X_100
uint FEE_PERCENT_TO_BUYBACK_X_100
uint REWARD_INTERVAL
uint ADMIN_CAN_CLAIM_AFTER
uint REWARD_RETURN_PERCENT_X_100
uint MIN_ETH_FEE_IN_WEI
address TRUSTED_DEPOSIT_TOKEN_ADDRESS
address TRUSTED_CTOKEN_ADDRESS
address TRUSTED_PLATFORM_TOKEN_ADDRESS
address BURN_ADDRESS
uint ONE_HUNDRED_X_100
uint contractStartTime
IUniswapV2Router uniswapRouterV2
EnumerableSet.AddressSet holders
address=>uint tokenBalances
address=>uint cTokenBalance
address=>uint depositTokenBalance
address=>uint totalTokensDepositedByUser
address=>uint totalTokensWithdrawnByUser
address=>uint totalEarnedCompoundDivs
address=>uint totalEarnedEthDivs
address=>uint totalEarnedTokenDivs
address=>uint totalEarnedPlatformTokenDivs
address=>uint depositTime
address=>uint lastClaimedTime
uint totalCTokens
uint totalDepositedTokens
uint POINT_MULTIPLIER
address=>uint lastTokenDivPoints
address=>uint tokenDivsBalance
uint totalTokenDivPoints
address=>uint lastEthDivPoints
address=>uint ethDivsBalance
uint totalEthDivPoints
address=>uint platformTokenDivsBalance
uint totalEthDisbursed
uint totalTokensDisbursed
__constructor__ ()
getNumberOfHolders()
tokenDivsOwing()
ethDivsOwing()
distributeEthDivs()
distributeTokenDivs()
getDepositorsList()
updateAccount()
platformTokenDivsOwing()
getEstimatedCompoundDivsOwing()
getConvertedBalance()
_claimEthDivs()
_claimTokenDivs()
_claimCompoundDivs()
_claimPlatformTokenDivs()
claimEthDivs()
claimTokenDivs()
claimCompoundDivs()
claimPlatformTokenDivs()
claim()
getExchangeRateCurrent()
getExchangeRateStored()
deposit()
withdraw()
emergencyWithdraw()
handleFee()
handleEthFee()
increaseTokenBalance()
decreaseTokenBalance()
addPlatformTokenBalance()
claimExtraTokens()
claimAnyToken()
IERC20
totalSupply()
balanceOf()
transfer()
allowance()
approve()
transferFrom()
CErc20
mint()
exchangeRateCurrent()
exchangeRateStored()
supplyRatePerBlock()
redeem()
redeemUnderlying()
for IERC20
for EnumerableSet.AddressSet
for uint
for address
IUniswapV2Router
WETH()
swapExactTokensForTokens()
swapExactETHForTokens()
getAmountsOut()
SafeERC20
SafeMath for uint256
Address for address
safeTransfer()
safeTransferFrom()
safeApprove()
safeIncreaseAllowance()
safeDecreaseAllowance()
_callOptionalReturn()
EnumerableSet
_add()
_remove()
_contains()
_length()
_at()
add()
remove()
contains()
length()
at()
ReentrancyGuard
uint256 _NOT_ENTERED
uint256 _ENTERED
uint256 _status
__constructor__ ()
Ownable
address _owner
__constructor__ ()
owner()
renounceOwnership()
transferOwnership()
for uint256
for address
SafeMath
add()
sub()
mul()
div()
mod()
Address
isContract()
sendValue()
functionCall()
functionCallWithValue()
_functionCallWithValue()

Slither Results

vault.sol

```
> slither vault.sol

INFO:Detectors:
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimTokenDivs() (vault.sol#1157)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault.sol#1085)
        - _claimCompoundDivs() (vault.sol#1158)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault.sol#1116)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimCompoundDivs() (vault.sol#1158)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        State variables written after the call(s):
        - _claimCompoundDivs() (vault.sol#1158)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault.sol#1020)
        - _claimCompoundDivs() (vault.sol#1158)
                - lastClaimedTime[account] = block.timestamp (vault.sol#1028)
        - _claimCompoundDivs() (vault.sol#1158)
                - lastEthDivPoints[account] = totalEthDivPoints (vault.sol#1018)
        - _claimCompoundDivs() (vault.sol#1158)
                - lastTokenDivPoints[account] = totalTokenDivPoints
(vault.sol#1012)
        - _claimCompoundDivs() (vault.sol#1158)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault.sol#1025)
        - _claimCompoundDivs() (vault.sol#1158)
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - _claimCompoundDivs() (vault.sol#1158)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault.sol#1014)
```

```
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimTokenDivs() (vault.sol#1157)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault.sol#1085)
        - _claimCompoundDivs() (vault.sol#1158)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault.sol#1116)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault.sol#1136)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimCompoundDivs() (vault.sol#1158)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        State variables written after the call(s):
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault.sol#1020)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - lastClaimedTime[account] = block.timestamp (vault.sol#1028)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - lastEthDivPoints[account] = totalEthDivPoints (vault.sol#1018)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - lastTokenDivPoints[account] = totalTokenDivPoints
(vault.sol#1012)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - platformTokenDivsBalance[msg.sender] = 0 (vault.sol#1125)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault.sol#1025)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
```

```
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault.sol#1014)
Reentrancy in Vault.withdraw(uint256,uint256,uint256,uint256) (vault.sol#1205-
1244):
        External calls:
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1217)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault.sol#1234)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0)
(vault.sol#1299)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault.sol#1300)
                -
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault.sol#1308)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1311)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1335)
        External calls sending eth:
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
        State variables written after the call(s):
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
```

Copyright 2021, Blockchain Consilium.

```
                - tokenBalances[token] = tokenBalances[token].add(amount)
(vault.sol#1344)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
                - totalEthDivPoints =
totalEthDivPoints.add(amount.mul(POINT_MULTIPLIER).div(totalDepositedTokens))
(vault.sol#965)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities
INFO:Detectors:
Vault._claimPlatformTokenDivs(uint256) (vault.sol#1122-1140) uses a dangerous
strict equality:
        - amount == 0 (vault.sol#1126)
Vault._claimTokenDivs() (vault.sol#1079-1089) uses a dangerous strict equality:
        - amount == 0 (vault.sol#1083)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#dangerous-strict-equalities
INFO:Detectors:
Reentrancy in Vault._claimCompoundDivs() (vault.sol#1090-1121):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault.sol#1092)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault.sol#1111)
Reentrancy in Vault.deposit(uint256,uint256,uint256) (vault.sol#1172-1204):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault.sol#1178)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault.sol#1183)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault.sol#1184)
        - require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).mint(amount) ==
0,mint failed!) (vault.sol#1187)
        State variables written after the call(s):
        - depositTokenBalance[msg.sender] =
depositTokenBalance[msg.sender].add(amount) (vault.sol#1195)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
Vault.deposit(uint256,uint256,uint256) (vault.sol#1172-1204) ignores return value
by holders.add(msg.sender) (vault.sol#1200)
Vault.withdraw(uint256,uint256,uint256,uint256) (vault.sol#1205-1244) ignores
return value by holders.remove(msg.sender) (vault.sol#1240)
Vault.emergencyWithdraw(uint256) (vault.sol#1247-1287) ignores return value by
holders.remove(msg.sender) (vault.sol#1283)
Vault.handleFee(uint256,uint256,uint256) (vault.sol#1289-1313) ignores return
value by
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault.sol#1308)
```

```
Vault.handleEthFee(uint256,uint256,uint256) (vault.sol#1315-1337) ignores return
value by uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-
return
INFO:Detectors:
Reentrancy in Vault._claimCompoundDivs() (vault.sol#1090-1121):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault.sol#1092)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
        State variables written after the call(s):
        - decreaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenRedeemed)
(vault.sol#1113)
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault.sol#1112)
        - totalTokensWithdrawnByUser[msg.sender] =
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault.sol#1115)
Reentrancy in Vault._claimCompoundDivs() (vault.sol#1090-1121):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault.sol#1092)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault.sol#1116)
        State variables written after the call(s):
        - totalEarnedCompoundDivs[msg.sender] =
totalEarnedCompoundDivs[msg.sender].add(depositTokenReceived) (vault.sol#1118)
Reentrancy in Vault._claimPlatformTokenDivs(uint256) (vault.sol#1122-1140):
        External calls:
        -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault.sol#1136)
        State variables written after the call(s):
        - totalEarnedPlatformTokenDivs[msg.sender] =
totalEarnedPlatformTokenDivs[msg.sender].add(estimatedAmountOut) (vault.sol#1137)
Reentrancy in Vault._claimTokenDivs() (vault.sol#1079-1089):
        External calls:
        - IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault.sol#1085)
        State variables written after the call(s):
        - totalEarnedTokenDivs[msg.sender] =
totalEarnedTokenDivs[msg.sender].add(amount) (vault.sol#1086)
Reentrancy in Vault.deposit(uint256,uint256,uint256) (vault.sol#1172-1204):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault.sol#1178)
        State variables written after the call(s):
```

```
        - totalTokensDepositedByUser[msg.sender] =
totalTokensDepositedByUser[msg.sender].add(amount) (vault.sol#1181)
Reentrancy in Vault.deposit(uint256,uint256,uint256) (vault.sol#1172-1204):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault.sol#1178)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault.sol#1183)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault.sol#1184)
        - require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).mint(amount) ==
0,mint failed!) (vault.sol#1187)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].add(cTokenReceived) (vault.sol#1191)
        - increaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenReceived)
(vault.sol#1193)
                - tokenBalances[token] = tokenBalances[token].add(amount)
(vault.sol#1344)
        - totalCTokens = totalCTokens.add(cTokenReceived) (vault.sol#1192)
        - totalDepositedTokens = totalDepositedTokens.add(amount)
(vault.sol#1196)
Reentrancy in Vault.deposit(uint256,uint256,uint256) (vault.sol#1172-1204):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault.sol#1178)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault.sol#1183)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault.sol#1184)
        - require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).mint(amount) ==
0,mint failed!) (vault.sol#1187)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1198)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1335)
        External calls sending eth:
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1198)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
        State variables written after the call(s):
```

```
        - depositTime[msg.sender] = block.timestamp (vault.sol#1201)
Reentrancy in Vault.emergencyWithdraw(uint256) (vault.sol#1247-1287):
        External calls:
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1259)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault.sol#1267)
        - decreaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenRedeemed)
(vault.sol#1269)
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault.sol#1268)
        - totalTokensWithdrawnByUser[msg.sender] =
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault.sol#1271)
Reentrancy in Vault.withdraw(uint256,uint256,uint256,uint256) (vault.sol#1205-
1244):
        External calls:
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1217)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault.sol#1225)
        - decreaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenRedeemed)
(vault.sol#1227)
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault.sol#1226)
        - totalTokensWithdrawnByUser[msg.sender] =
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault.sol#1229)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in Vault._claimCompoundDivs() (vault.sol#1090-1121):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault.sol#1092)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault.sol#1116)
        Event emitted after the call(s):
        - CompoundRewardClaimed(msg.sender,depositTokenReceived) (vault.sol#1120)
Reentrancy in Vault._claimPlatformTokenDivs(uint256) (vault.sol#1122-1140):
        External calls:
        -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault.sol#1136)
        Event emitted after the call(s):
        - PlatformTokenRewardClaimed(msg.sender,estimatedAmountOut)
(vault.sol#1139)
Reentrancy in Vault._claimTokenDivs() (vault.sol#1079-1089):
        External calls:
```

```
        - IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault.sol#1085)
        Event emitted after the call(s):
        - TokenRewardClaimed(msg.sender,amount) (vault.sol#1088)
Reentrancy in Vault.addPlatformTokenBalance(uint256) (vault.sol#1350-1355):
        External calls:
        -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),a
mount) (vault.sol#1352)
        Event emitted after the call(s):
        - PlatformTokenAdded(amount) (vault.sol#1354)
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimTokenDivs() (vault.sol#1157)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault.sol#1085)
        - _claimCompoundDivs() (vault.sol#1158)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault.sol#1116)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimCompoundDivs() (vault.sol#1158)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        Event emitted after the call(s):
        - CompoundRewardClaimed(msg.sender,depositTokenReceived) (vault.sol#1120)
                - _claimCompoundDivs() (vault.sol#1158)
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimTokenDivs() (vault.sol#1157)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault.sol#1085)
        - _claimCompoundDivs() (vault.sol#1158)
                - exchangeRateCurrent =
CErc20(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault.sol#1163)
```

Copyright 2021, Blockchain Consilium.

```
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1103)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault.sol#1116)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault.sol#1136)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimCompoundDivs() (vault.sol#1158)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        Event emitted after the call(s):
        - PlatformTokenRewardClaimed(msg.sender,estimatedAmountOut)
(vault.sol#1139)
                - _claimPlatformTokenDivs(_amountOutMin_platformTokens)
(vault.sol#1159)
Reentrancy in Vault.deposit(uint256,uint256,uint256) (vault.sol#1172-1204):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault.sol#1178)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault.sol#1183)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault.sol#1184)
        - require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).mint(amount) ==
0,mint failed!) (vault.sol#1187)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1198)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
```

Copyright 2021, Blockchain Consilium.

```
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1335)
        External calls sending eth:
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1198)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
        Event emitted after the call(s):
        - Deposit(msg.sender,amount) (vault.sol#1203)
        - EtherRewardDisbursed(amount) (vault.sol#968)
                - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1198)
Reentrancy in Vault.emergencyWithdraw(uint256) (vault.sol#1247-1287):
        External calls:
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1259)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault.sol#1276)
        Event emitted after the call(s):
        - Withdraw(msg.sender,depositTokenReceived) (vault.sol#1286)
Reentrancy in Vault.withdraw(uint256,uint256,uint256,uint256) (vault.sol#1205-
1244):
        External calls:
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1217)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault.sol#1234)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0)
(vault.sol#1299)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault.sol#1300)
                -
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault.sol#1308)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1311)
        External calls sending eth:
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        Event emitted after the call(s):
```

Copyright 2021, Blockchain Consilium.

```
        - TokenRewardDisbursed(amount) (vault.sol#975)
                - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
Reentrancy in Vault.withdraw(uint256,uint256,uint256,uint256) (vault.sol#1205-
1244):
        External calls:
        -
require(bool,string)(CErc20(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault.sol#1217)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault.sol#1234)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0)
(vault.sol#1299)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault.sol#1300)
                -
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault.sol#1308)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1311)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault.sol#1335)
        External calls sending eth:
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline)
(vault.sol#1236)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline)
(vault.sol#1332)
        Event emitted after the call(s):
        - EtherRewardDisbursed(amount) (vault.sol#968)
                - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault.sol#1237)
        - Withdraw(msg.sender,depositTokenReceived) (vault.sol#1243)
```

Copyright 2021, Blockchain Consilium.

```
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Vault.updateAccount(address) (vault.sol#1009-1029) uses timestamp for comparisons
        Dangerous comparisons:
        - platformTokensOwing > 0 (vault.sol#1024)
Vault.platformTokenDivsOwing(address) (vault.sol#1031-1054) uses timestamp for
comparisons
        Dangerous comparisons:
        - _now > stakingEndTime (vault.sol#1038)
        - lastClaimedTime[account] >= _now (vault.sol#1042)
Vault._claimPlatformTokenDivs(uint256) (vault.sol#1122-1140) uses timestamp for
comparisons
        Dangerous comparisons:
        - amount == 0 (vault.sol#1126)
        - require(bool,string)(estimatedAmountOut >=
_amountOutMin_platformTokens,_claimPlatformTokenDivs: slippage error!)
(vault.sol#1134)
Vault.withdraw(uint256,uint256,uint256,uint256) (vault.sol#1205-1244) uses
timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp.sub(depositTime[msg.sender]) >
LOCKUP_DURATION,You recently deposited, please wait before withdrawing.)
(vault.sol#1208)
Vault.emergencyWithdraw(uint256) (vault.sol#1247-1287) uses timestamp for
comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp.sub(depositTime[msg.sender]) >
LOCKUP_DURATION,You recently deposited, please wait before withdrawing.)
(vault.sol#1250)
Vault.claimAnyToken(address,uint256) (vault.sol#1367-1374) uses timestamp for
comparisons
        Dangerous comparisons:
        - require(bool,string)(now >
contractStartTime.add(ADMIN_CAN_CLAIM_AFTER),Contract not expired yet!)
(vault.sol#1368)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp
INFO:Detectors:
Address.isContract(address) (vault.sol#182-191) uses assembly
        - INLINE ASM (vault.sol#189)
Address._functionCallWithValue(address,bytes,uint256,string) (vault.sol#275-296)
uses assembly
        - INLINE ASM (vault.sol#288-291)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-
usage
INFO:Detectors:
solc-0.6.12 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (vault.sol#209-215):
        - (success) = recipient.call{value: amount}() (vault.sol#213)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string)
(vault.sol#275-296):
        - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-
level-calls
```

```
INFO:Detectors:
Function IUniswapV2Router.WETH() (vault.sol#819) is not in mixedCase
Parameter Vault.getConvertedBalance(uint256)._cTokenBalance (vault.sol#1062) is
not in mixedCase
Parameter Vault.claimPlatformTokenDivs(uint256)._amountOutMin_platformTokens
(vault.sol#1151) is not in mixedCase
Parameter Vault.claim(uint256)._amountOutMin_platformTokens (vault.sol#1155) is
not in mixedCase
Parameter Vault.deposit(uint256,uint256,uint256)._amountOutMin_ethFeeBuyBack
(vault.sol#1172) is not in mixedCase
Parameter
Vault.withdraw(uint256,uint256,uint256,uint256)._amountOutMin_ethFeeBuyBack
(vault.sol#1205) is not in mixedCase
Parameter
Vault.withdraw(uint256,uint256,uint256,uint256)._amountOutMin_tokenFeeBuyBack
(vault.sol#1205) is not in mixedCase
Parameter Vault.handleFee(uint256,uint256,uint256)._amountOutMin_tokenFeeBuyBack
(vault.sol#1289) is not in mixedCase
Parameter Vault.handleEthFee(uint256,uint256,uint256)._amountOutMin_ethFeeBuyBack
(vault.sol#1315) is not in mixedCase
Constant Vault.uniswapRouterV2 (vault.sol#884) is not in
UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Reentrancy in Vault._claimEthDivs() (vault.sol#1068-1078):
        External calls:
        - msg.sender.transfer(amount) (vault.sol#1074)
        State variables written after the call(s):
        - totalEarnedEthDivs[msg.sender] =
totalEarnedEthDivs[msg.sender].add(amount) (vault.sol#1075)
        Event emitted after the call(s):
        - EtherRewardClaimed(msg.sender,amount) (vault.sol#1077)
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        State variables written after the call(s):
        - _claimTokenDivs() (vault.sol#1157)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault.sol#1020)
        - _claimTokenDivs() (vault.sol#1157)
                - lastClaimedTime[account] = block.timestamp (vault.sol#1028)
        - _claimTokenDivs() (vault.sol#1157)
                - lastEthDivPoints[account] = totalEthDivPoints (vault.sol#1018)
        - _claimTokenDivs() (vault.sol#1157)
                - lastTokenDivPoints[account] = totalTokenDivPoints
(vault.sol#1012)
        - _claimTokenDivs() (vault.sol#1157)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault.sol#1025)
        - _claimTokenDivs() (vault.sol#1157)
```

Copyright 2021, Blockchain Consilium.

```
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - _claimTokenDivs() (vault.sol#1157)
                - tokenDivsBalance[msg.sender] = 0 (vault.sol#1082)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault.sol#1014)
        - _claimTokenDivs() (vault.sol#1157)
                - totalEarnedTokenDivs[msg.sender] =
totalEarnedTokenDivs[msg.sender].add(amount) (vault.sol#1086)
        Event emitted after the call(s):
        - TokenRewardClaimed(msg.sender,amount) (vault.sol#1088)
                - _claimTokenDivs() (vault.sol#1157)
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimCompoundDivs() (vault.sol#1158)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        State variables written after the call(s):
        - _claimCompoundDivs() (vault.sol#1158)
                - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault.sol#1111)
        - _claimCompoundDivs() (vault.sol#1158)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault.sol#1020)
        - _claimCompoundDivs() (vault.sol#1158)
                - lastClaimedTime[account] = block.timestamp (vault.sol#1028)
        - _claimCompoundDivs() (vault.sol#1158)
                - lastEthDivPoints[account] = totalEthDivPoints (vault.sol#1018)
        - _claimCompoundDivs() (vault.sol#1158)
                - lastTokenDivPoints[account] = totalTokenDivPoints
(vault.sol#1012)
        - _claimCompoundDivs() (vault.sol#1158)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault.sol#1025)
        - _claimCompoundDivs() (vault.sol#1158)
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - _claimCompoundDivs() (vault.sol#1158)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault.sol#1014)
        - _claimCompoundDivs() (vault.sol#1158)
                - totalCTokens = totalCTokens.sub(cTokenRedeemed)
(vault.sol#1112)
        - _claimCompoundDivs() (vault.sol#1158)
                - totalEarnedCompoundDivs[msg.sender] =
totalEarnedCompoundDivs[msg.sender].add(depositTokenReceived) (vault.sol#1118)
        - _claimCompoundDivs() (vault.sol#1158)
                - totalTokensWithdrawnByUser[msg.sender] =
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault.sol#1115)
        Event emitted after the call(s):
        - CompoundRewardClaimed(msg.sender,depositTokenReceived) (vault.sol#1120)
```

```
                - _claimCompoundDivs() (vault.sol#1158)
Reentrancy in Vault.claim(uint256) (vault.sol#1155-1160):
        External calls:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        External calls sending eth:
        - _claimEthDivs() (vault.sol#1156)
                - msg.sender.transfer(amount) (vault.sol#1074)
        - _claimTokenDivs() (vault.sol#1157)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimCompoundDivs() (vault.sol#1158)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault.sol#279)
        State variables written after the call(s):
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault.sol#1020)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - lastClaimedTime[account] = block.timestamp (vault.sol#1028)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - lastEthDivPoints[account] = totalEthDivPoints (vault.sol#1018)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - lastTokenDivPoints[account] = totalTokenDivPoints
(vault.sol#1012)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - platformTokenDivsBalance[msg.sender] = 0 (vault.sol#1125)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault.sol#1025)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - tokenBalances[token] = tokenBalances[token].sub(amount)
(vault.sol#1347)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault.sol#1014)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault.sol#1159)
                - totalEarnedPlatformTokenDivs[msg.sender] =
totalEarnedPlatformTokenDivs[msg.sender].add(estimatedAmountOut) (vault.sol#1137)
        Event emitted after the call(s):
        - PlatformTokenRewardClaimed(msg.sender,estimatedAmountOut)
(vault.sol#1139)
                - _claimPlatformTokenDivs(_amountOutMin_platformTokens)
(vault.sol#1159)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable Vault.cTokenBalance (vault.sol#919) is too similar to Vault.tokenBalances
(vault.sol#916)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-
names-are-too-similar
INFO:Detectors:
Vault.slitherConstructorConstantVariables() (vault.sol#849-1375) uses literals
with too many digits:
        - MIN_ETH_FEE_IN_WEI = 400000 * 1 * 10 ** 9 (vault.sol#867)
Vault.slitherConstructorConstantVariables() (vault.sol#849-1375) uses literals
with too many digits:
        - BURN_ADDRESS = 0x000000000000000000000000000000000000dEaD
(vault.sol#873)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits
INFO:Detectors:
owner() should be declared external:
        - Ownable.owner() (vault.sol#769-771)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (vault.sol#788-791)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (vault.sol#797-801)
getNumberOfHolders() should be declared external:
        - Vault.getNumberOfHolders() (vault.sol#911-913)
getDepositorsList(uint256,uint256) should be declared external:
        - Vault.getDepositorsList(uint256,uint256) (vault.sol#982-1007)
getEstimatedCompoundDivsOwing(address) should be declared external:
        - Vault.getEstimatedCompoundDivsOwing(address) (vault.sol#1056-1060)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
INFO:Slither:vault.sol analyzed (10 contracts with 72 detectors), 65 result(s)
found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and
Github integration
```

vault-weth.sol

```
> slither vault-weth.sol

INFO:Detectors:
Reentrancy in VaultWETH._claimCompoundDivs() (vault-weth.sol#1098-1131):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault-weth.sol#1100)
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
        -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault-weth.sol#1121)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
        External calls:
        - _claimTokenDivs() (vault-weth.sol#1166)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount) (vault-
weth.sol#1093)
        - _claimCompoundDivs() (vault-weth.sol#1167)
```

```
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault-weth.sol#1126)
        External calls sending eth:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        State variables written after the call(s):
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault-weth.sol#1028)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - lastClaimedTime[account] = block.timestamp (vault-
weth.sol#1036)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - lastEthDivPoints[account] = totalEthDivPoints (vault-
weth.sol#1026)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - lastTokenDivPoints[account] = totalTokenDivPoints (vault-
weth.sol#1020)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault-weth.sol#1033)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - tokenBalances[token] = tokenBalances[token].sub(amount) (vault-
weth.sol#1362)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault-weth.sol#1022)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
        External calls:
        - _claimTokenDivs() (vault-weth.sol#1166)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount) (vault-
weth.sol#1093)
        - _claimCompoundDivs() (vault-weth.sol#1167)
```

Copyright 2021, Blockchain Consilium.

```
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault-weth.sol#1126)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault-weth.sol#1145)
        External calls sending eth:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        State variables written after the call(s):
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault-weth.sol#1028)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - lastClaimedTime[account] = block.timestamp (vault-
weth.sol#1036)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - lastEthDivPoints[account] = totalEthDivPoints (vault-
weth.sol#1026)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - lastTokenDivPoints[account] = totalTokenDivPoints (vault-
weth.sol#1020)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - platformTokenDivsBalance[msg.sender] = 0 (vault-weth.sol#1135)
```

Copyright 2021, Blockchain Consilium.

```
                - platformTokenDivsBalance[account] = 
platformTokenDivsBalance[account].add(platformTokensOwing) (vault-weth.sol#1033)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - tokenBalances[token] = tokenBalances[token].sub(amount) (vault-
weth.sol#1362)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - tokenDivsBalance[account] = 
tokenDivsBalance[account].add(tokensOwing) (vault-weth.sol#1022)
Reentrancy in VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-
1216):
        External calls:
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault-weth.sol#1187)
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0) 
(vault-weth.sol#1192)
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount) 
(vault-weth.sol#1193)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).withdraw(amount) (vault-
weth.sol#1197)
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        External calls sending eth:
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        State variables written after the call(s):
        - depositTokenBalance[msg.sender] = 
depositTokenBalance[msg.sender].add(amount) (vault-weth.sol#1207)
Reentrancy in VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-
1216):
        External calls:
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault-weth.sol#1187)
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0) 
(vault-weth.sol#1192)
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount) 
(vault-weth.sol#1193)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).withdraw(amount) (vault-
weth.sol#1197)
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data) 
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value: 
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
```

```
                 -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1350)
        External calls sending eth:
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
        State variables written after the call(s):
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - totalEthDivPoints =
totalEthDivPoints.add(amount.mul(POINT_MULTIPLIER).div(totalDepositedTokens))
(vault-weth.sol#973)
Reentrancy in VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-
weth.sol#1217-1258):
        External calls:
        -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1229)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1236)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault-weth.sol#1248)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0)
(vault-weth.sol#1315)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault-weth.sol#1316)
                -
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault-weth.sol#1323)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1326)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1236)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        State variables written after the call(s):
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
```

```
                - tokenBalances[token] = tokenBalances[token].add(amount) (vault-
weth.sol#1359)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - totalTokenDivPoints =
totalTokenDivPoints.add(amount.mul(POINT_MULTIPLIER).div(totalDepositedTokens))
(vault-weth.sol#980)
Reentrancy in VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-
weth.sol#1217-1258):
        External calls:
        -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1229)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1236)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault-weth.sol#1248)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0)
(vault-weth.sol#1315)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault-weth.sol#1316)
                -
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault-weth.sol#1323)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1326)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1251)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1350)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1236)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1251)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
```

Copyright 2021, Blockchain Consilium.

```
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
        State variables written after the call(s):
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1251)
                - tokenBalances[token] = tokenBalances[token].add(amount) (vault-
weth.sol#1359)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1251)
                - totalEthDivPoints =
totalEthDivPoints.add(amount.mul(POINT_MULTIPLIER).div(totalDepositedTokens))
(vault-weth.sol#973)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities
INFO:Detectors:
VaultWETH._claimPlatformTokenDivs(uint256) (vault-weth.sol#1132-1149) uses a
dangerous strict equality:
        - amount == 0 (vault-weth.sol#1136)
VaultWETH._claimTokenDivs() (vault-weth.sol#1087-1097) uses a dangerous strict
equality:
        - amount == 0 (vault-weth.sol#1091)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#dangerous-strict-equalities
INFO:Detectors:
VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-1216) ignores
return value by holders.add(msg.sender) (vault-weth.sol#1212)
VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-weth.sol#1217-1258)
ignores return value by holders.remove(msg.sender) (vault-weth.sol#1254)
VaultWETH.emergencyWithdraw(uint256) (vault-weth.sol#1261-1303) ignores return
value by holders.remove(msg.sender) (vault-weth.sol#1299)
VaultWETH.handleFee(uint256,uint256,uint256) (vault-weth.sol#1305-1328) ignores
return value by
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault-weth.sol#1323)
VaultWETH.handleEthFee(uint256,uint256,uint256) (vault-weth.sol#1330-1352) ignores
return value by uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-
return
INFO:Detectors:
Reentrancy in VaultWETH._claimCompoundDivs() (vault-weth.sol#1098-1131):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault-weth.sol#1100)
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
        -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        State variables written after the call(s):
        - decreaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenRedeemed) (vault-
weth.sol#1123)
```

Copyright 2021, Blockchain Consilium.

```
                - tokenBalances[token] = tokenBalances[token].sub(amount) (vault-
weth.sol#1362)
        - totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault-weth.sol#1122)
        - totalTokensWithdrawnByUser[msg.sender] =
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault-
weth.sol#1125)
Reentrancy in VaultWETH._claimCompoundDivs() (vault-weth.sol#1098-1131):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault-weth.sol#1100)
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
        -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault-weth.sol#1126)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        State variables written after the call(s):
        - totalEarnedCompoundDivs[msg.sender] =
totalEarnedCompoundDivs[msg.sender].add(depositTokenReceived) (vault-
weth.sol#1128)
Reentrancy in VaultWETH._claimPlatformTokenDivs(uint256) (vault-weth.sol#1132-
1149):
        External calls:
        -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault-weth.sol#1145)
        State variables written after the call(s):
        - totalEarnedPlatformTokenDivs[msg.sender] =
totalEarnedPlatformTokenDivs[msg.sender].add(estimatedAmountOut) (vault-
weth.sol#1146)
Reentrancy in VaultWETH._claimTokenDivs() (vault-weth.sol#1087-1097):
        External calls:
        - IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault-weth.sol#1093)
        State variables written after the call(s):
        - totalEarnedTokenDivs[msg.sender] =
totalEarnedTokenDivs[msg.sender].add(amount) (vault-weth.sol#1094)
Reentrancy in VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-
1216):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault-weth.sol#1187)
        State variables written after the call(s):
        - totalTokensDepositedByUser[msg.sender] =
totalTokensDepositedByUser[msg.sender].add(amount) (vault-weth.sol#1190)
Reentrancy in VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-
1216):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault-weth.sol#1187)
```

```
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault-weth.sol#1192)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault-weth.sol#1193)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).withdraw(amount) (vault-
weth.sol#1197)
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        External calls sending eth:
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].add(cTokenReceived) (vault-weth.sol#1203)
        - increaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenReceived) (vault-
weth.sol#1205)
                - tokenBalances[token] = tokenBalances[token].add(amount) (vault-
weth.sol#1359)
        - totalCTokens = totalCTokens.add(cTokenReceived) (vault-weth.sol#1204)
        - totalDepositedTokens = totalDepositedTokens.add(amount) (vault-
weth.sol#1208)
Reentrancy in VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-
1216):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault-weth.sol#1187)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault-weth.sol#1192)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault-weth.sol#1193)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).withdraw(amount) (vault-
weth.sol#1197)
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1350)
        External calls sending eth:
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
```

```
                - uniswapRouterV2.swapExactETHForTokens{value: buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-weth.sol#1347)
        State variables written after the call(s):
        - depositTime[msg.sender] = block.timestamp (vault-weth.sol#1213)
Reentrancy in VaultWETH.emergencyWithdraw(uint256) (vault-weth.sol#1261-1303):
        External calls:
        - 
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) == 0,redeemUnderlying failed!) (vault-weth.sol#1273)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: depositTokenReceived}() (vault-weth.sol#1280)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: depositTokenReceived}() (vault-weth.sol#1280)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] = 
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault-weth.sol#1283)
        - decreaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenRedeemed) (vault-weth.sol#1285)
                - tokenBalances[token] = tokenBalances[token].sub(amount) (vault-weth.sol#1362)
        - totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault-weth.sol#1284)
        - totalTokensWithdrawnByUser[msg.sender] = 
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault-weth.sol#1287)
Reentrancy in VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-weth.sol#1217-1258):
        External calls:
        - 
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) == 0,redeemUnderlying failed!) (vault-weth.sol#1229)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: depositTokenReceived}() (vault-weth.sol#1236)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: depositTokenReceived}() (vault-weth.sol#1236)
        State variables written after the call(s):
        - cTokenBalance[msg.sender] = 
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault-weth.sol#1239)
        - decreaseTokenBalance(TRUSTED_CTOKEN_ADDRESS,cTokenRedeemed) (vault-weth.sol#1241)
                - tokenBalances[token] = tokenBalances[token].sub(amount) (vault-weth.sol#1362)
        - totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault-weth.sol#1240)
        - totalTokensWithdrawnByUser[msg.sender] = 
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault-weth.sol#1243)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in VaultWETH._claimCompoundDivs() (vault-weth.sol#1098-1131):
        External calls:
        - exchangeRateCurrent = getExchangeRateCurrent() (vault-weth.sol#1100)
                - exchangeRateCurrent = 
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
        - 
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) == 0,redeemUnderlying failed!) (vault-weth.sol#1111)
```

```
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault-weth.sol#1126)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        Event emitted after the call(s):
        - CompoundRewardClaimed(msg.sender,depositTokenReceived) (vault-
weth.sol#1130)
Reentrancy in VaultWETH._claimPlatformTokenDivs(uint256) (vault-weth.sol#1132-
1149):
        External calls:
        -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault-weth.sol#1145)
        Event emitted after the call(s):
        - PlatformTokenRewardClaimed(msg.sender,estimatedAmountOut) (vault-
weth.sol#1148)
Reentrancy in VaultWETH._claimTokenDivs() (vault-weth.sol#1087-1097):
        External calls:
        - IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount)
(vault-weth.sol#1093)
        Event emitted after the call(s):
        - TokenRewardClaimed(msg.sender,amount) (vault-weth.sol#1096)
Reentrancy in VaultWETH.addPlatformTokenBalance(uint256) (vault-weth.sol#1365-
1370):
        External calls:
        -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),a
mount) (vault-weth.sol#1367)
        Event emitted after the call(s):
        - PlatformTokenAdded(amount) (vault-weth.sol#1369)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
        External calls:
        - _claimTokenDivs() (vault-weth.sol#1166)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount) (vault-
weth.sol#1093)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault-weth.sol#1126)
```

```
        External calls sending eth:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        Event emitted after the call(s):
        - CompoundRewardClaimed(msg.sender,depositTokenReceived) (vault-
weth.sol#1130)
                - _claimCompoundDivs() (vault-weth.sol#1167)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
        External calls:
        - _claimTokenDivs() (vault-weth.sol#1166)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,amount) (vault-
weth.sol#1093)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - exchangeRateCurrent =
CEther(TRUSTED_CTOKEN_ADDRESS).exchangeRateCurrent() (vault-weth.sol#1172)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1111)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
) (vault-weth.sol#1126)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(msg.sender,estimatedAmountOut)
(vault-weth.sol#1145)
        External calls sending eth:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
```

Copyright 2021, Blockchain Consilium.

```
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        Event emitted after the call(s):
        - PlatformTokenRewardClaimed(msg.sender,estimatedAmountOut) (vault-
weth.sol#1148)
                - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
Reentrancy in VaultWETH.deposit(uint256,uint256,uint256) (vault-weth.sol#1181-
1216):
        External calls:
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransferFrom(msg.sender,address(this),am
ount) (vault-weth.sol#1187)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,0)
(vault-weth.sol#1192)
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(TRUSTED_CTOKEN_ADDRESS,amount)
(vault-weth.sol#1193)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).withdraw(amount) (vault-
weth.sol#1197)
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1350)
        External calls sending eth:
        - CEther(TRUSTED_CTOKEN_ADDRESS).mint{value: amount}() (vault-
weth.sol#1198)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1210)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
        Event emitted after the call(s):
        - Deposit(msg.sender,amount) (vault-weth.sol#1215)
        - EtherRewardDisbursed(amount) (vault-weth.sol#976)
                - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault-weth.sol#1210)
Reentrancy in VaultWETH.emergencyWithdraw(uint256) (vault-weth.sol#1261-1303):
        External calls:
        -
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) ==
0,redeemUnderlying failed!) (vault-weth.sol#1273)
```

```
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: 
depositTokenReceived}() (vault-weth.sol#1280)
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault-weth.sol#1292)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: 
depositTokenReceived}() (vault-weth.sol#1280)
        Event emitted after the call(s):
        - Withdraw(msg.sender,depositTokenReceived) (vault-weth.sol#1302)
Reentrancy in VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-
weth.sol#1217-1258):
        External calls:
        - 
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) == 
0,redeemUnderlying failed!) (vault-weth.sol#1229)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: 
depositTokenReceived}() (vault-weth.sol#1236)
        - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault-weth.sol#1248)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data) 
(vault-weth.sol#279)
                - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0) 
(vault-weth.sol#1315)
                - 
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault-weth.sol#1316)
                - 
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault-weth.sol#1323)
                - 
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1326)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: 
depositTokenReceived}() (vault-weth.sol#1236)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - (success,returndata) = target.call{value: weiValue}(data) 
(vault-weth.sol#279)
        Event emitted after the call(s):
        - TokenRewardDisbursed(amount) (vault-weth.sol#983)
                - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) 
(vault-weth.sol#1250)
Reentrancy in VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-
weth.sol#1217-1258):
        External calls:
        - 
require(bool,string)(CEther(TRUSTED_CTOKEN_ADDRESS).redeemUnderlying(amount) == 
0,redeemUnderlying failed!) (vault-weth.sol#1229)
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value: 
depositTokenReceived}() (vault-weth.sol#1236)
```

```
        -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeTransfer(msg.sender,depositTokenReceived
AfterFee) (vault-weth.sol#1248)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),0)
(vault-weth.sol#1315)
                -
IERC20(TRUSTED_DEPOSIT_TOKEN_ADDRESS).safeApprove(address(uniswapRouterV2),buyBack
FeeAmount) (vault-weth.sol#1316)
                -
uniswapRouterV2.swapExactTokensForTokens(buyBackFeeAmount,_amountOutMin_tokenFeeBu
yBack,path,address(this),deadline) (vault-weth.sol#1323)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1326)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1251)
                - returndata = address(token).functionCall(data,SafeERC20: low-
level call failed) (vault-weth.sol#599)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
                -
IERC20(TRUSTED_PLATFORM_TOKEN_ADDRESS).safeTransfer(BURN_ADDRESS,platformTokensRec
eived) (vault-weth.sol#1350)
        External calls sending eth:
        - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1236)
        - handleFee(feeAmount,_amountOutMin_tokenFeeBuyBack,deadline) (vault-
weth.sol#1250)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline) (vault-
weth.sol#1251)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - uniswapRouterV2.swapExactETHForTokens{value:
buyBackFeeAmount}(_amountOutMin_ethFeeBuyBack,path,address(this),deadline) (vault-
weth.sol#1347)
        Event emitted after the call(s):
        - EtherRewardDisbursed(amount) (vault-weth.sol#976)
                - handleEthFee(msg.value,_amountOutMin_ethFeeBuyBack,deadline)
(vault-weth.sol#1251)
        - Withdraw(msg.sender,depositTokenReceived) (vault-weth.sol#1257)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
VaultWETH.updateAccount(address) (vault-weth.sol#1017-1037) uses timestamp for
comparisons
        Dangerous comparisons:
        - platformTokensOwing > 0 (vault-weth.sol#1032)
```

```
VaultWETH.platformTokenDivsOwing(address) (vault-weth.sol#1039-1062) uses
timestamp for comparisons
        Dangerous comparisons:
        - _now > stakingEndTime (vault-weth.sol#1046)
        - lastClaimedTime[account] >= _now (vault-weth.sol#1050)
VaultWETH._claimPlatformTokenDivs(uint256) (vault-weth.sol#1132-1149) uses
timestamp for comparisons
        Dangerous comparisons:
        - amount == 0 (vault-weth.sol#1136)
        - require(bool,string)(estimatedAmountOut >=
_amountOutMin_platformTokens,_claimPlatformTokenDivs: slippage error!) (vault-
weth.sol#1143)
VaultWETH.withdraw(uint256,uint256,uint256,uint256) (vault-weth.sol#1217-1258)
uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp.sub(depositTime[msg.sender]) >
LOCKUP_DURATION,You recently deposited, please wait before withdrawing.) (vault-
weth.sol#1220)
VaultWETH.emergencyWithdraw(uint256) (vault-weth.sol#1261-1303) uses timestamp for
comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp.sub(depositTime[msg.sender]) >
LOCKUP_DURATION,You recently deposited, please wait before withdrawing.) (vault-
weth.sol#1264)
VaultWETH.claimAnyToken(address,uint256) (vault-weth.sol#1382-1389) uses timestamp
for comparisons
        Dangerous comparisons:
        - require(bool,string)(now >
contractStartTime.add(ADMIN_CAN_CLAIM_AFTER),Contract not expired yet!) (vault-
weth.sol#1383)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp
INFO:Detectors:
Address.isContract(address) (vault-weth.sol#182-191) uses assembly
        - INLINE ASM (vault-weth.sol#189)
Address._functionCallWithValue(address,bytes,uint256,string) (vault-weth.sol#275-
296) uses assembly
        - INLINE ASM (vault-weth.sol#288-291)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-
usage
INFO:Detectors:
solc-0.6.12 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (vault-weth.sol#209-215):
        - (success) = recipient.call{value: amount}() (vault-weth.sol#213)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string)
(vault-weth.sol#275-296):
        - (success,returndata) = target.call{value: weiValue}(data) (vault-
weth.sol#279)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-
level-calls
INFO:Detectors:
Function IUniswapV2Router.WETH() (vault-weth.sol#819) is not in mixedCase
Parameter VaultWETH.getConvertedBalance(uint256)._cTokenBalance (vault-
weth.sol#1070) is not in mixedCase
Parameter VaultWETH.claimPlatformTokenDivs(uint256)._amountOutMin_platformTokens
(vault-weth.sol#1160) is not in mixedCase
```

```
Parameter VaultWETH.claim(uint256)._amountOutMin_platformTokens (vault-
weth.sol#1164) is not in mixedCase
Parameter VaultWETH.deposit(uint256,uint256,uint256)._amountOutMin_ethFeeBuyBack
(vault-weth.sol#1181) is not in mixedCase
Parameter
VaultWETH.withdraw(uint256,uint256,uint256,uint256)._amountOutMin_ethFeeBuyBack
(vault-weth.sol#1217) is not in mixedCase
Parameter
VaultWETH.withdraw(uint256,uint256,uint256,uint256)._amountOutMin_tokenFeeBuyBack
(vault-weth.sol#1217) is not in mixedCase
Parameter
VaultWETH.handleFee(uint256,uint256,uint256)._amountOutMin_tokenFeeBuyBack (vault-
weth.sol#1305) is not in mixedCase
Parameter
VaultWETH.handleEthFee(uint256,uint256,uint256)._amountOutMin_ethFeeBuyBack
(vault-weth.sol#1330) is not in mixedCase
Constant VaultWETH.uniswapRouterV2 (vault-weth.sol#882) is not in
UPPER_CASE_WITH_UNDERSCORES
Variable VaultWETH.TRUSTED_DEPOSIT_TOKEN_ADDRESS (vault-weth.sol#887) is not in
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Reentrancy in VaultWETH._claimEthDivs() (vault-weth.sol#1076-1086):
        External calls:
        - msg.sender.transfer(amount) (vault-weth.sol#1082)
        State variables written after the call(s):
        - totalEarnedEthDivs[msg.sender] =
totalEarnedEthDivs[msg.sender].add(amount) (vault-weth.sol#1083)
        Event emitted after the call(s):
        - EtherRewardClaimed(msg.sender,amount) (vault-weth.sol#1085)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
        External calls:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        External calls sending eth:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        State variables written after the call(s):
        - _claimTokenDivs() (vault-weth.sol#1166)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault-weth.sol#1028)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - lastClaimedTime[account] = block.timestamp (vault-
weth.sol#1036)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - lastEthDivPoints[account] = totalEthDivPoints (vault-
weth.sol#1026)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - lastTokenDivPoints[account] = totalTokenDivPoints (vault-
weth.sol#1020)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault-weth.sol#1033)
        - _claimTokenDivs() (vault-weth.sol#1166)
```

```
		- tokenBalances[token] = tokenBalances[token].sub(amount) (vault-
weth.sol#1362)
	- _claimTokenDivs() (vault-weth.sol#1166)
		- tokenDivsBalance[msg.sender] = 0 (vault-weth.sol#1090)
		- tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault-weth.sol#1022)
	- _claimTokenDivs() (vault-weth.sol#1166)
		- totalEarnedTokenDivs[msg.sender] =
totalEarnedTokenDivs[msg.sender].add(amount) (vault-weth.sol#1094)
	Event emitted after the call(s):
	- TokenRewardClaimed(msg.sender,amount) (vault-weth.sol#1096)
		- _claimTokenDivs() (vault-weth.sol#1166)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
	External calls:
	- _claimEthDivs() (vault-weth.sol#1165)
		- msg.sender.transfer(amount) (vault-weth.sol#1082)
	External calls sending eth:
	- _claimEthDivs() (vault-weth.sol#1165)
		- msg.sender.transfer(amount) (vault-weth.sol#1082)
	- _claimTokenDivs() (vault-weth.sol#1166)
		- (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
		- IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
	State variables written after the call(s):
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- cTokenBalance[msg.sender] =
cTokenBalance[msg.sender].sub(cTokenRedeemed) (vault-weth.sol#1121)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault-weth.sol#1028)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- lastClaimedTime[account] = block.timestamp (vault-
weth.sol#1036)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- lastEthDivPoints[account] = totalEthDivPoints (vault-
weth.sol#1026)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- lastTokenDivPoints[account] = totalTokenDivPoints (vault-
weth.sol#1020)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault-weth.sol#1033)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- tokenBalances[token] = tokenBalances[token].sub(amount) (vault-
weth.sol#1362)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault-weth.sol#1022)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- totalCTokens = totalCTokens.sub(cTokenRedeemed) (vault-
weth.sol#1122)
	- _claimCompoundDivs() (vault-weth.sol#1167)
		- totalEarnedCompoundDivs[msg.sender] =
totalEarnedCompoundDivs[msg.sender].add(depositTokenReceived) (vault-
weth.sol#1128)
```

```
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - totalTokensWithdrawnByUser[msg.sender] =
totalTokensWithdrawnByUser[msg.sender].add(depositTokenReceived) (vault-
weth.sol#1125)
        Event emitted after the call(s):
        - CompoundRewardClaimed(msg.sender,depositTokenReceived) (vault-
weth.sol#1130)
                - _claimCompoundDivs() (vault-weth.sol#1167)
Reentrancy in VaultWETH.claim(uint256) (vault-weth.sol#1164-1169):
        External calls:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        External calls sending eth:
        - _claimEthDivs() (vault-weth.sol#1165)
                - msg.sender.transfer(amount) (vault-weth.sol#1082)
        - _claimTokenDivs() (vault-weth.sol#1166)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        - _claimCompoundDivs() (vault-weth.sol#1167)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
                - IWETH(TRUSTED_DEPOSIT_TOKEN_ADDRESS).deposit{value:
depositTokenReceived}() (vault-weth.sol#1118)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - (success,returndata) = target.call{value: weiValue}(data)
(vault-weth.sol#279)
        State variables written after the call(s):
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - ethDivsBalance[account] = ethDivsBalance[account].add(weiOwing)
(vault-weth.sol#1028)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - lastClaimedTime[account] = block.timestamp (vault-
weth.sol#1036)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - lastEthDivPoints[account] = totalEthDivPoints (vault-
weth.sol#1026)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - lastTokenDivPoints[account] = totalTokenDivPoints (vault-
weth.sol#1020)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - platformTokenDivsBalance[msg.sender] = 0 (vault-weth.sol#1135)
                - platformTokenDivsBalance[account] =
platformTokenDivsBalance[account].add(platformTokensOwing) (vault-weth.sol#1033)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - tokenBalances[token] = tokenBalances[token].sub(amount) (vault-
weth.sol#1362)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
                - tokenDivsBalance[account] =
tokenDivsBalance[account].add(tokensOwing) (vault-weth.sol#1022)
        - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
```

```
                - totalEarnedPlatformTokenDivs[msg.sender] =
totalEarnedPlatformTokenDivs[msg.sender].add(estimatedAmountOut) (vault-
weth.sol#1146)
        Event emitted after the call(s):
        - PlatformTokenRewardClaimed(msg.sender,estimatedAmountOut) (vault-
weth.sol#1148)
                - _claimPlatformTokenDivs(_amountOutMin_platformTokens) (vault-
weth.sol#1168)
Reference: https://github.com/crytic/slither/wiki/Detector-
Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable VaultWETH.cTokenBalance (vault-weth.sol#927) is too similar to
VaultWETH.tokenBalances (vault-weth.sol#924)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-
names-are-too-similar
INFO:Detectors:
VaultWETH.slitherConstructorConstantVariables() (vault-weth.sol#855-1390) uses
literals with too many digits:
        - MIN_ETH_FEE_IN_WEI = 400000 * 1 * 10 ** 9 (vault-weth.sol#873)
VaultWETH.slitherConstructorConstantVariables() (vault-weth.sol#855-1390) uses
literals with too many digits:
        - BURN_ADDRESS = 0x000000000000000000000000000000000000dEaD (vault-
weth.sol#878)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits
INFO:Detectors:
owner() should be declared external:
        - Ownable.owner() (vault-weth.sol#769-771)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (vault-weth.sol#788-791)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (vault-weth.sol#797-801)
getNumberOfHolders() should be declared external:
        - VaultWETH.getNumberOfHolders() (vault-weth.sol#919-921)
getDepositorsList(uint256,uint256) should be declared external:
        - VaultWETH.getDepositorsList(uint256,uint256) (vault-weth.sol#990-1015)
getEstimatedCompoundDivsOwing(address) should be declared external:
        - VaultWETH.getEstimatedCompoundDivsOwing(address) (vault-weth.sol#1064-
1068)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
INFO:Slither:vault-weth.sol analyzed (11 contracts with 72 detectors), 68
result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and
Github integration
```