ApyFlow V2

Smart Contract Security Audit

Prepared by ShellBoxes

Oct 14th, 2022 - Oct 19th, 2022

Shellboxes.com

contact@shellboxes.com

Document Properties

Client	ApyFlow
Version	1.0
Classification	Public

Scope

Repo	Commit Hash
https://gitlab.com/apyflowcom/apyflow-dlt	9c1be4f9d883e2778d3b016e0a5b8b7864da1c7d

Re-Audit

Repo	Commit Hash
https://gitlab.com/apyflowcom/apyflow-dlt	d08184ab88696cccf6f1a9e4188f49aad2a1f6a3

Contacts

COMPANY	EMAIL
ShellBoxes	contact@shellboxes.com

Contents

1 Introduction

ApyFlow engaged ShellBoxes to conduct a security assessment on the ApyFlow V2 beginning on Oct 14th, 2022 and ending Oct 19th, 2022. In this report, we detail our methodical approach to evaluate potential security issues associated with the implementation of smart contracts, by exposing possible semantic discrepancies between the smart contract code and design document, and by recommending additional ideas to optimize the existing code. Our findings indicate that the current version of smart contracts can still be enhanced further due to the presence of many security and performance concerns.

This document summarizes the findings of our audit.

1.1 About ApyFlow

Apyflow automates investing in DeFi protocols by using smart contracts for different blockchains and DeFi protocols. The main goal and feature of the product are to make investing in DeFi easy and understandable, even for those who do not have experience in Blockchain and cryptocurrencies at all.

Issuer	ApyFlow
Website	`https://apyflow.com/`
Type	Solidity Smart Contract
Audit Method	Whitebox

1.2 Approach & Methodology

ShellBoxes used a combination of manual and automated security testing to achieve a balance between efficiency, timeliness, practicability, and correctness within the audit's scope. While manual testing is advised for identifying problems in logic, procedure, and implementation, automated testing techniques help to expand the coverage of smart contracts and can quickly detect code that does not comply with security best practices.

1.2.1 Risk Methodology

Vulnerabilities or bugs identified by ShellBoxes are ranked using a risk assessment technique that considers both the LIKELIHOOD and IMPACT of a security incident. This framework is effective at conveying the features and consequences of technological vulnerabilities.

Its quantitative paradigm enables repeatable and precise measurement, while also revealing the underlying susceptibility characteristics that were used to calculate the Risk scores. A risk level will be assigned to each vulnerability on a scale of 5 to 1, with 5 indicating the greatest possibility or impact.

- Likelihood quantifies the probability of a certain vulnerability being discovered and exploited in the untamed.
- Impact quantifies the technical and economic costs of a successful attack.
- Severity indicates the risk's overall criticality.

Probability and impact are classified into three categories: H, M, and L, which correspond to high, medium, and low, respectively. Severity is determined by probability and impact and is categorized into four levels, namely Critical, High, Medium, and Low.

Impact	Likelihood: High	Likelihood: Medium	Likelihood: Low
High	Critical	High	Medium
Medium	High	Medium	Low
Low	Medium	Low	Low

2 Findings Overview

2.1 Summary

The following is a synopsis of our conclusions from our analysis of the ApyFlow V2 implementation. During the first part of our audit, we examine the smart contract source code and run the codebase via a static code analyzer. The objective here is to find known coding problems statically and then manually check (reject or confirm) issues highlighted by the tool. Additionally, we check business logics, system processes, and DeFi-related components manually to identify potential hazards and/or defects.

2.2 Key Findings

In general, these smart contracts are well-designed and constructed, but their implementation might be improved by addressing the discovered flaws, which include 4 critical-severity, 1 high-severity, 2 medium-severity, 13 low-severity, 7 undetermined-severity vulnerabilities.

Vulnerabilities	Severity	Status
A.1. Unprotected Call Can Lead To A Drain Of LINK Funds	CRITICAL	Mitigated
B.1. User Can Deposit Any ERC20 Asset	CRITICAL	Fixed
C.1. CurveConverter's swap Function Not Protected	CRITICAL	Fixed
D.1. UniswapV2Converter's swap Function Not Protected	CRITICAL	Fixed
A.2. Power Centralization In The Score Update	HIGH	Mitigated
C.2. Missing Transfer Verification	MEDIUM	Fixed
D.2. Possible DoS In The swap Function Of UniswapV2Converter	MEDIUM	Fixed
A.3. Missing Address Verification	LOW	Mitigated
A.4. Missing Value Verification	LOW	Mitigated
B.2. Missing Address Verification	LOW	Fixed
B.3. Floating Pragma	LOW	Fixed

D.3. Missing Address Verification	LOW	Fixed
E.1. Missing Address Verification	LOW	Mitigated
F.1. Missing Address Verification	LOW	Fixed
F.2. Missing Value Verification	LOW	Fixed
F.3. Loss Precision	LOW	Acknowledged
G.1. Possible Race Condition Can Lead To Unexpected Behavior	LOW	Acknowledged
G.2. Loss Precision	LOW	Acknowledged
H.1. Missing Value Verification	LOW	Mitigated
I.1. Loss Precision	LOW	Fixed
C.3. Missing Functionnality In The Code	UNDETERMINED	Fixed
F.4. Allowance Not Revoked After Removing The Vault	UNDETERMINED	Not Fixed
G.3. Rebalance Algorithm Not Guaranteed To Be Called	UNDETERMINED	Acknowledged
G.4. Allowance Not Revoked After Removing The Vault	UNDETERMINED	Not Fixed
G.5. Dynamic Decimal Hardcoded	UNDETERMINED	Fixed
J.1. Allowance Not Revoked After modifying the converter	UNDETERMINED	Not Fixed
K.1. Dynamic Decimal Hardcoded	UNDETERMINED	Acknowledged

3 Finding Details

A PortfolioScoreOracle.sol

A.1 Unprotected Call Can Lead To A Drain Of LINK Funds [CRITICAL]

Description:

In the requestVaultData function located in the PortfolioScoreOracle any user can call this function to trigger the process of getting the score value from the API, however this can lead to a drain of funds if a malicious user intentionally called the function multiple time.

Code:

Listing 1: PortfolioScoreOracle.sol

```
39 function requestVaultData(address vaultAddress) public returns (bytes32
       ↪ requestId)
40 {
41 ....
```

Risk Level:

Likelihood – 5
Impact - 4

Recommendation:

Consider adding access control to the requestVaultData function.

Status - Mitigated

The team has decided to remove the PortofolioScoreOracle contract and avoid all the interactions from Chainlink, therefore the portofolio score will be updated by the team onchain.

A.2 Power Centralization In The Score Update [HIGH]

Description:

The scoreData, profiltScore and apyInPpm are updated using ChainLink oracle, the oracle is calling external APIs existing in ApyFlow servers, this presents an important risk of centralization and if somehow an attacker managed to access to ApyFlow server he can change these values.

Code:

Listing 2: PortfolioScoreOracle.sol

```
39 function requestVaultData(address vaultAddress) public returns (bytes32
       ↪ requestId)
40  {
41   Chainlink.Request memory request = buildChainlinkRequest(jobId,
       ↪ address(this), this.fulfill.selector);
42   string memory vault = toAsciiString(vaultAddress);
43   string memory url = string.concat(uri, vault);
44
45   // Set the URL to perform the GET request on
46   request.add("get", url);
47
48   request.add("path", "risc_score,0");
49   request.add("path", "risc_score,1");
50   request.add("path", "risc_score,2");
51   request.add("path", "risc_score,3");
52   request.add("path", "risc_score,4");
53   request.add("path", "risc_score,5");
54   request.add("path", "profit_score");
55   request.add("path", "apy");
56
57   // Sends the request
58   requestId = sendChainlinkRequestTo(oracle, request, fee);
59   vaultForRequestId[requestId] = vaultAddress;
```

```
61    emit DataRequested(url, requestId);
62  }
```

Risk Level:

Likelihood – 2
Impact - 5

Recommendation:

Consider adding a logic of a time lock that will not update immediately this value, but wait a certain period of time. The behavior can also be documented and explained to the user.

Status - Mitigated

The team has decided to remove the PortofolioScoreOracle contract and avoid all the interactions from Chainlink, therefore the portofolio score will be updated by the team onchain.

A.3 Missing Address Verification [LOW]

Description:

Certain functions lack a safety check in the address, the address-type argument should include a zero-address test, otherwise, the contract's functionality may become inaccessible or tokens may be burned in perpetuity.
In the constructor of PortfolioScoreOracle, the contract should verify if _oracle is different than address(0) .

Code:

Listing 3: PortfolioScoreOracle.sol

```
22  constructor (address _oracle, bytes32 _jobId, uint256 _fee, address
        ↪ _link, string memory _uri)
23  {
```

```
24    if (_link == address(0))
25    {
26     setPublicChainlinkToken();
27    }
28    else
29    {
30     setChainlinkToken(_link);
31    }

33    oracle = _oracle;
34    jobId = _jobId;
35    fee = _fee;
36    uri = _uri;
37   }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

It is recommended to make sure the addresses provided in the arguments are different than the address(0).

Status - Mitigated

The team has decided to remove the PortofolioScoreOracle contract and avoid all the interactions from Chainlink, therefore the portofolio score will be updated by the team onchain.

A.4 Missing Value Verification [LOW]

Description:

Certain functions lack a safety check in the values, the values of the arguments should be verified to allow only the ones that go with the contract's logic.
In the constructor of PortfolioScoreOracle, the contract should verify if _fee is less than 100%.

Code:

Listing 4: PortfolioScoreOracle.sol

```
35    fee = _fee;
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

Consider verifying if _fee is less than 100%.

Status - Mitigated

The team has decided to remove the PortofolioScoreOracle contract and avoid all the interactions from Chainlink, therefore the portofolio score will be updated by the team onchain.

B ApyFlowZap.sol

B.1 User Can Deposit Any ERC20 Asset [CRITICAL]

Description:

In the deposit function located in the ApyFlowZap a user can deposit any ERC20 token and gain shares on it, this can have serious impact if the user crafted malicious ERC20 tokens.
- Same logic follows with the redeem function, the user can redeem any ERC20 token.

Code:

Listing 5: ApyFlowZap.sol

```
28 function deposit(address token, uint value) external returns(uint256
       ↪ shares)
29 {
30         IERC20(token).safeTransferFrom(msg.sender, address(this), value);
31         uint256[] memory amounts = new uint256[](apyflow.vaultsLength());
32         uint256 totalPortfolioScore = apyflow.totalPortfolioScore();
33         if (IERC20(token).allowance(address(this), address(assetConverter
               ↪ )) < value) {
34             IERC20(token).safeIncreaseAllowance(address(assetConverter),
                   ↪ type(uint256).max);
35         }
36         for (uint i = 0; i < amounts.length; i++) {
37             SingleAssetVault vault = SingleAssetVault(apyflow.getVault(i)
                   ↪ );
38             address tokenToDeposit = vault.asset();
39             uint256 amountToDeposit = vault.totalPortfolioScore() * value
                   ↪  / totalPortfolioScore;
40             if (tokenToDeposit != token)
41                 amounts[i] = assetConverter.swap(token, tokenToDeposit,
                       ↪ amountToDeposit);
42             else
43                 amounts[i] = amountToDeposit;
```

```
            if (IERC20(tokenToDeposit).allowance(address(this), address(
                ↪ apyflow)) < amounts[i]) {
                IERC20(tokenToDeposit).safeIncreaseAllowance(address(
                    ↪ apyflow), type(uint256).max);
            }
        }
        shares = apyflow.deposit(amounts, msg.sender);
    }
```

Listing 6: ApyFlowZap.sol

```
    function redeem(address token, uint shares) external returns(uint256
        ↪ assets) {
        apyflow.safeTransferFrom(msg.sender, address(this), shares);
        uint256[] memory amounts = apyflow.redeem(shares, address(this));
        for (uint i = 0; i < amounts.length; i++) {
            if (amounts[i] > 0) {
                address withdrawnToken = SingleAssetVault(apyflow.getVault
                    ↪ (i)).asset();
                if (withdrawnToken != token) {
                    if (IERC20(withdrawnToken).allowance(address(this),
                        ↪ address(assetConverter)) < amounts[i]) {
                        IERC20(withdrawnToken).safeIncreaseAllowance(
                            ↪ address(assetConverter), type(uint256).max);
                    }
                    assets += assetConverter.swap(withdrawnToken, token,
                        ↪ amounts[i]);
                } else {
                    assets += amounts[i];
                }
            }
        }
        IERC20(token).safeTransfer(msg.sender, assets);
    }
```

Risk Level:

Likelihood – 5
Impact - 5

Recommendation:

Add a verification to ensure that the asset is supported by the protocol.

Status - Fixed

The team resolved the issue by adding the tokenAllowed modifier which is responsible for ensuring if the token is authorized.

B.2 Missing Address Verification [LOW]

Description:

Certain functions lack a safety check in the address, the address-type argument should include a zero-address test, otherwise, the contract's functionality may become inaccessible or tokens may be burned in perpetuity.
In the constructor of ApyFlowZap, the contract should verify if _apyflow is different than address(0) .

Code:

Listing 7: ApyFlowZap.sol

```
23      constructor(ApyFlow _apyflow) {
24          apyflow = _apyflow;
25          assetConverter = apyflow.assetConverter();
26      }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

It is recommended to make sure the addresses provided in the arguments are different from the address(0).

Status - Fixed

The team resolved the issue by validating if _apyflow is different from address(0).

B.3 Floating Pragma [LOW]

Description:

The contract makes use of the floating-point pragma 0.8.0. Contracts should be deployed using the same compiler version. Locking the pragma helps ensure that contracts are not unintentionally deployed using another pragma, such as an obsolete version, that may introduce issues in the contract system.

Code:

Listing 8: ApyFlowZap.sol

```
1  pragma solidity >=0.8.0;
```

Risk Level:

Likelihood – 1
Impact - 1

Recommendation:

Consider locking the pragma version. It is advised that floating pragma should not be used in production. Both truffle-config.js and hardhat.config.js support locking the pragma version.

Status - Fixed

The team resolved the issue by locking the pragma version to 0.8.15.

C CurveConverter.sol

C.1 CurveConverter's swap Function Not Protected [CRITICAL]

Description:

The CurveConverter contract contains a swap function, this function takes the following parameters (source,destination,value,beneficiary), however this function is not protected and anyone can call it.

Code:

Listing 9: CurveConverter.sol

```
35 function swap(address source, address destination, uint256 value,
       ↪ address beneficiary) external returns (uint256)
36 {
37         .....
38 }
```

Risk Level:

Likelihood – 5
Impact - 5

Recommendation:

Consider adding an access control to the swap function to reduce the risk.

Status - Fixed

The team resolved the issue by verifying if the caller is the assetConverter.

C.2 Missing Transfer Verification [MEDIUM]

Description:

The ERC20 standard token implementation functions return the transaction status as a Boolean. It is a good practice to check for the return status of the function call to ensure that the transaction was executed successfully. It is the developer's responsibility to enclose these function calls with require() to ensure that, when the intended ERC20 function call returns false, the caller transaction also fails.

Code:

Listing 10: CurveConverter.sol

```
41    IERC20(destination).transfer(beneficiary, result);
```

Risk Level:

Likelihood – 3
Impact - 3

Recommendation:

Use the safeTransfer function from the safeERC20 Implementation, or put the transfer call inside an assert or require to verify that it returned true.

Status - Fixed

The team resolved the issue by using safeTransfer from the safeERC20 OZ library.

C.3 Missing Functionnality In The Code [UNDETERMINED]

Description:

In the CurveConverter, the contract only contains the swap function, however it's mentioned that another function called stop should exist.

Code:

Listing 11: CurveConverter.sol

```
33  // add stop function
```

Recommendation:

Consider removing it from the comments if the stop function is not necessary, or implementing the stop function.

Status - Fixed

The team resolved the issue by removing the comment.

D UniswapV2Converter.sol

D.1 UniswapV2Converter's swap Function Not Protected [CRITICAL]

Description:

The UniswapV2Converter contract contains a swap function, this function takes the following parameters (source,destination,value,beneficiary), however this function is not protected and anyone can call it.

Code:

Listing 12: UniswapV2Converter.sol

```
49      function swap(
50          address source,
51          address destination,
52          uint256 value,
53          address beneficiary
54      ) external returns (uint256) {
55      .......
56      }
```

Risk Level:

Likelihood – 5
Impact - 5

Recommendation:

Consider adding an access control to the swap function to reduce the risk.

Status - Fixed

The team resolved the issue by verifying if the caller is the assetConverter.

D.2 Possible DoS In The swap Function Of UniswapV2Converter [MEDIUM]

Description:

In the swap function of the UniswapV2Converter contract, the contract is verifying if we have an existent liquidity pool using the .getPair from the Uniswap contract if it's not the case, the path is automatically changed to have two liquidity pools composed of WETH and the other asset, however nothing guarantees that we will have those liquidity pools and the swap function can be blocked.

Code:

Listing 13: UniswapV2Converter.sol

```
49    function swap(
50        address source,
51        address destination,
52        uint256 value,
53        address beneficiary
54     ) external returns (uint256) {
55        address[] memory path;
56        if (factory.getPair(source, destination) != address(0)) {
57            path = new address[](2);
58            path[0] = source;
59            path[1] = destination;
60        } else {
61            path = new address[](3);
62            path[0] = source;
63            path[1] = address(WETH);
64            path[2] = destination;
65        }
```

Risk Level:

Likelihood – 4

Impact - 2

Recommendation:

Consider Verifying the existence of the two liquidity pools using the .getPair function.

Listing 14: UniswapV2Converter.sol

```
49  if(getPair(source,WETH)==address(0) or getPair(destination,WETH)==0))
50    {
51        revert();
52    }
```

Status - Fixed

The team resolved the issue by verifying if the liquidity pools exist before doing any operations.

D.3 Missing Address Verification [LOW]

Description:

Certain functions lack a safety check in the address, the address-type argument should include a zero-address test, otherwise, the contract's functionality may become inaccessible or tokens may be burned in perpetuity.
In the constructor of UniswapV2Converter, the contract should verify if _router is different from address(0) .

Code:

Listing 15: UniswapV2Converter.sol

```
43      constructor(address _router) Ownable() {
44          router = IUniswapV2Router(_router);
45          factory = IUniswapV2Factory(router.factory());
46          WETH = IERC20(router.WETH());
47      }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

It is recommended to make sure the addresses provided in the arguments are different from the address(0).

Status - Fixed

The team resolved the issue by validating if _router is different from address(0).

E UniswapV3Converter.sol

E.1 Missing Address Verification [LOW]

Description:

Certain functions lack a safety check in the address, the address-type argument should include a zero-address test, otherwise, the contract's functionality may become inaccessible or tokens may be burned in perpetuity.
In the constructor of UniswapV3Converter, the contract should verify if _router is different from address(0) .

Code:

Listing 16: UniswapV3Converter.sol

```
32      constructor(address _router) Ownable() {
33          router = IUniswapV3(_router);
34      }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

It is recommended to make sure the addresses provided in the arguments are different from the address(0).

Status - Mitigated

The team resolved the issue by removing the UniswapV3Converter contract.

F SingleAssetVault.sol

F.1 Missing Address Verification [LOW]

Description:

Certain functions lack a safety check in the address, the address-type argument should include a zero-address test, otherwise, the contract's functionality may become inaccessible or tokens may be burned in perpetuity.
In the constructor of SingleAssetVault, the contract should verify if addressForFees is different from address(0) .

Code:

Listing 17: SingleAssetVault.sol

```
33 constructor (address portfolioScore, IERC20Metadata asset_, string
     ↪ memory name, string memory symbol, address addressForFees,
     ↪ uint256 _fee) ERC4626(asset_) ERC20(name, symbol)
34 {
35  require(portfolioScore != address(0), "Zero address provided");

37  oracle = PortfolioScore(portfolioScore);
38  feeTreasury = addressForFees;
39  feeInPpm = _fee;
40  _decimals = asset_.decimals();
41  lastPricePerShare = 10 ** decimals();
42 }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

It is recommended to make sure the addresses provided in the arguments are different from the address(0).

Status - Fixed

The team resolved the issue by validating if addressForFees is different from address(0).

F.2 Missing Value Verification [LOW]

Description:

Certain functions lack a safety check in the values, the values of the arguments should be verified to allow only the ones that go with the contract's logic.
In the constructor of SingleAssetVault, the contract should verify if _fee is less than 100% .

Code:

Listing 18: SingleAssetVault.sol

```
39        feeInPpm = _fee;
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

Consider verifying if _fee is less than 100%.

Status - Fixed

The team resolved the issue by verifying that _fee is less than 1000.

F.3 Loss Precision [LOW]

Description:

In the recomputePricePerShareAndHarvestFee function, the fee variable is calculated using the following formula :
fee = profit * feeInPpm / 1000;
If the profit * feeInPpm is less than 1000 the fee will be equal to 0.

Code:

Listing 19: SingleAssetVault.sol

```
78      uint256 fee = profit * feeInPpm / 1000;
```

Risk Level:

Likelihood – 3
Impact - 2

Recommendation:

Consider verifying that profit * feeInPpm is greater than 1000

Status - Acknowledged

The team acknowledged the risk, and they are considering that the amount which can be saved by the user by exploiting this vulnerability and not paying the fees is much lower than the transaction cost for such a small deposit.

F.4 Allowance Not Revoked After Removing The Vault [UNDETERMINED]

Description:

When removing the Vault using the removeVault function located in the SingleAssetVault, the allowance of the vault is not revoked.

Code:

Listing 20: SingleAssetVault.sol

```
51     function removeVault(address vaultAddress) external onlyOwner
52  {
53   require(vaults.contains(vaultAddress), "vault does not exist");
54   IERC4626 vault = IERC4626(vaultAddress);
55   vault.withdraw(vault.convertToAssets(vault.balanceOf(address(this))),
        ↪ address(this), address(this));
56   vaults.remove(vaultAddress);
57  }
```

Recommendation:

Consider changing the allowance of the vault to 0 when removing the vault.

Status - Not Fixed

The team added a verification of the allowance when adding the vault, however this won't solve the issue and the old vault will still have allowance.

G ApyFlow.sol

G.1 Possible Race Condition Can Lead To Unexpected Behavior [LOW]

Description:

When the user wants to deposit using the deposit function located in the ApyFlow contract, the user will only provide a list of amounts depending on each vault, for example we have three vaults [v1,v2,v3] the user will send [100,500,1800] and we understand that the equivalent amount of v2 is 500. However, this process can be affected by a race condition. If the owner removed a vault and added another vault and this transaction is executed before the deposit function, the user will have to deposit in an unexpected vault. In the example, we can imagine that the admin removed v3 and added v4 therefore the list of vaults will be the following [v1,v2,v4] and the equivalent amount of v4 will be 1800.

Code:

Listing 21: ApyFlow.sol

```
41    constructor(address _converter, address _oracle)
42        ERC20("ApyFlow", "APYFLW")
43    {
44        require(_converter != address(0), "Zero address provided");
45        require(_oracle != address(0), "Zero address provided");
46        assetConverter = AssetConverter(_converter);
47        oracle = PortfolioScore(_oracle);
48    }
```

Risk Level:

Likelihood – 1

Impact - 5

Recommendation:

Consider adding an array of the vaults that the user is willing to deposit in as a parameter of the deposit function.

Status - Acknowledged

The team acknowledged the risk stating that they are not going to add/remove SingleAsset-Vaults very often.

G.2 Loss Precision [LOW]

Description:

In the _verifyAmounts function, the value (deviation * 100) / totalAmount should be less than or equal to 10, However if the deviation * 100 is less than totalAmount the result will be equal to 0.
Same issue in the deposit function.

Code:

Listing 22: ApyFlow.sol

```
131         require((deviation * 100) / totalAmount <= 10, "Invalid amounts")
                ↪ ;
```

Listing 23: ApyFlow.sol

```
148         uint256 shares = convertToShares(
149                 (vaultAssets * (10**decimals())) /
150                     (10**vault.decimals())
151             );
```

Risk Level:

Likelihood – 3
Impact - 2

Recommendation:

Consider changing this formula require((deviation * 100) / totalAmount <= 10, "Invalid amounts");
to the following : require(deviation * 10 <= totalAmount, "Invalid amounts");

Status - Acknowledged

The team acknowledged the risk and stated that they are not planning to set such hard restrictions.

G.3 Rebalance Algorithm Not Guaranteed To Be Called [UNDETERMINED]

Description:

The rebalance algorithm is a fund transfer between different protocols and blockchains to increase the effective rate. In the documentation it's written that Users' funds are automatically distributed to protocols according to a rebalancing algorithm , however this function is not called automatically and needs to be manually triggered.

Code:

Listing 24: ApyFlow.sol

```
220     function rebalance(
221         address sourceVaultAddress,
222         address destinationVaultAddress,
223         uint256 assets
224      ) external {
225     require(vaults.contains(sourceVaultAddress));
226     require(vaults.contains(destinationVaultAddress));
```

Recommendation:

Consider calling this function in each deposit/withdraw or document this behavior and exactly how this process is triggered automatically.

Status - Acknowledged

The team acknowledged the risk and stated that the rebalance function is called by their off chain script, and they will provide this information in the white paper.

G.4 Allowance Not Revoked After Removing The Vault [UNDETERMINED]

Description:

When removing the Vault using the removeVault function located in the ApyFlow, the allowance of the vault is not revoked.

Code:

Listing 25: ApyFlow.sol

```
95  function removeVault(address vault) external onlyOwner {
96   require(vaults.contains(vault), "The vault is not added");
97   vaults.remove(vault);
98  }
```

Recommendation:

Consider changing the allowance of the vault to 0 when removing the vault.

Status - Not Fixed

The team added a verification of the allowance when adding the vault, however this won't solve the issue and the old vault will still have allowance.

G.5 Dynamic Decimal Hardcoded [UNDETERMINED]

Description:

In pricePerToken function located in the ApyFlow contract.It calculates the totalAmount based on the amounts. However, it's multiplied with (10**18), the formula can be correct however it's not advised to have this value hard-coded in the code.
Same issue in the _verifyAmounts function.

Code:

Listing 26: ApyFlow.sol

```
76      function pricePerToken() public view returns (uint256) {
77          return convertToAssets(10**18);
78      }
```

Listing 27: ApyFlow.sol

```
117   totalAmount +=
118     (amounts[i] * (10**18)) /
119     (10**SingleAssetVault(vaults.at(i)).decimals());
```

Listing 28: ApyFlow.sol

```
124   uint256 expected = (totalAmount *
125     vault.totalPortfolioScore() *
126     10**(vault.decimals())) /
127     (totalScore * (10**18));
```

Recommendation:

Consider changing the 18 value with the decimals() annotation.

Status - Fixed

The team resolved the issue by not hard-coding the 18 value and replacing it with the decimals() annotation.

H PortfolioScore.sol

H.1 Missing Value Verification [LOW]

Description:

Certain functions lack a safety check in the values, the values of the arguments should be verified to allow only the ones that align with the contract's logic.
In the getPortfolioScore function, the contract should verify that 3*riskScore + 2*scoreData.profitScore is less than 50000, this is dictated by the fact that the number of ppms should be between 0 and 1000.

Code:

Listing 29: PortfolioScore.sol

```
35    return (riskScore * 60 / 100 + scoreData.profitScore * 40) / 100;
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

Consider verifying return of getPortfolioScore is between 0 and 10000.

Status - Mitigated

The team mitigated the issue by removing all the calculations from the PortofolioScore contract.

I WrappedERC4626CurvePoolConvex.sol

I.1 Loss Precision [LOW]

Description:

In the _convertLpAmountToShares function, the shares variable is calculated using the following formula :
shares = (lpAmount * totalSupply()) / balance;
If the lpAmount * totalSupply() is less than balance the shares will be equal to 0.

Code:

Listing 30: WrappedERC4626CurvePoolConvex.sol

```
132   shares = (lpAmount * totalSupply()) / balance;
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

Consider verifying that lpAmount * totalSupply() is greater than balance.

Status - Fixed

The team resolved the issue by verifying if shares is greater than 0.

J AssetConverter.sol

J.1 Allowance Not Revoked After modifying the converter [UNDETERMINED]

Description:

When modifying the converter using the updateConverter function located in the AssetConverter, the allowance of the old converter is not revoked

Code:

Listing 31: AssetConverter.sol

```
26  function updateConverter(address source, address destination, address
        ↪ newConverter) external onlyOwner
27  {
28   if (IERC20(source).allowance(address(this), newConverter) == 0) {
29    IERC20(source).safeIncreaseAllowance(newConverter, type(uint256).max)
          ↪ ;
30   }
31   converters[source][destination] = IConverter(newConverter);
32  }
```

Recommendation:

Consider changing the allowance of the old converter to 0 when updating it.

Status - Not Fixed

The team added a verification of the allowance when updating the converter, however this won't solve the issue and the old vault will still have allowance.

K WrappedERC4626CurveMetapoolConvex.sol

K.1 Dynamic Decimal Hardcoded [UNDETERMINED]

Description:

In _convertToShares function, the contract is calculating the metapoolLpAmount based on the assets. However, we are multiplying this value by (10**18), the formula can be correct however it's not advised to have this value hard-coded in the code.

Code:

Listing 32: WrappedERC4626CurveMetapoolConvex.sol

```
175  assets = (assets * (10**18)) / (10**depositTokenDecimals);
```

Recommendation:

Consider changing the 18 value with the decimals() annotation.

Status - Acknowledged

The team acknowledged the risk and stated that they are using 10**18 hard-coded because the Curve's get_virtual_price method always returns values normalized by 10**18.

4 Best Practices

BP.1 Remove empty constructor

Description:

No need to have an empty constructor in the PortofiloScore contract.

Code:

Listing 33: PortofiloScore.sol

```
17    constructor ()
18   {
19
20   }
```

BP.2 Remove dead code

Description:

Remove the dead code from the getPortofolioScore function located in the PortofiloScore contract.

Code:

Listing 34: PortofiloScore.sol

```
33    //riskScore /= 100;
```

5 Tests

Results:

```
Brownie v1.19.2 - Python development framework for Ethereum

============== test session starts
    ↪ ======================================
platform win32 -- Python 3.9.13, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: C:\Users\inas\Downloads\apyflow-dlt-9
    ↪ c1be4f9d883e2778d3b016e0a5b8b7864da1c7d\apyflow-dlt-9
    ↪ c1be4f9d883e2778d3b016e0a5b8b7864da1c7d
plugins: eth-brownie-1.19.2, anyio-3.5.0, hypothesis-6.27.3, forked
    ↪ -1.4.0, xdist-1.34.0, web3-5.31.1
collected 47 items

Launching 'ganache-cli.cmd --port 8545 --gasLimit 12000000 --accounts 10
    ↪  --hardfork istanbul --mnemonic brownie'...

tests\for_local\test_deploy.py .... [ 8%]
tests\for_local\erc4626_wrappers\test_totalAsstets_of_yearn.py .. [ 12%]
tests\for_local\multi-asset\test_convert_to_assets.py . [ 14%]
tests\for_local\multi-asset\test_deposit_to_apyflow.py .. [ 19%]
tests\for_local\multi-asset\test_rebalance.py . [ 21%]
tests\for_local\single_asset_vault\test_deposit.py .... [ 29%]
tests\for_local\single_asset_vault\test_recompute_pps_amd_harvest_fee.py
    ↪  .. [ 34%]
tests\for_local\single_asset_vault\test_totalAssets.py .. [ 38%]
tests\for_local\single_asset_vault\test_withdraw.py .... [ 46%]
tests\for_local\test_mocks\test_converter.py .... [ 55%]
tests\for_local\test_mocks\test_yearn.py ..... [ 68%]
tests\for_mainnet_fork\test_asset_converter.py EE [ 72%]
tests\for_mainnet_fork\test_curve_convex_meta_vaults.py EEE [ 78%]
tests\for_mainnet_fork\test_curve_convex_vaults.py EEE [ 85%]
```

```
tests\for_mainnet_fork\test_curve_vault.py EE [ 89%]
tests\for_mainnet_fork\test_single_asset_vaults.py EE [ 93%]
tests\for_mainnet_fork\test_yearn_v2_vaults.py EEE [100%]

============ ERRORS ============================================
_____________________________________ ERROR at setup of
    ↪ test_uniswap_v2 _____________________________________

ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def cvx(ERC20, network_config):
>   return ERC20.at(network_config["cvx"])
E   KeyError: 'cvx'

tests\for_mainnet_fork\conftest.py:59: KeyError
_________________ERROR at setup of test_uniswap_v3
    ↪ ______________________

ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}

    @pytest.fixture(scope="module")
    def weth(ERC20, network_config):
>   return ERC20.at(network_config["weth"])
E   KeyError: 'weth'

tests\for_mainnet_fork\conftest.py:39: KeyError
```

```
__________ ERROR at setup of test_deposit
    ↪ ______________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
> return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E KeyError: 'sushiswap_router'

tests\for_mainnet_fork\conftest.py:19: KeyError
__________ ERROR at setup of test_compound ________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
> return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E KeyError: 'sushiswap_router'
```

```
tests\for_mainnet_fork\conftest.py:19: KeyError
__________ ERROR at setup of test_withdraw ___________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}

    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
>   return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E   KeyError: 'sushiswap_router'

tests\for_mainnet_fork\conftest.py:19: KeyError
_________ ERROR at setup of test_deposit ______________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
>   return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E   KeyError: 'sushiswap_router'
```

```
tests\for_mainnet_fork\conftest.py:19: KeyError
_________ ERROR at setup of test_compound ____________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
>   return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E   KeyError: 'sushiswap_router'

tests\for_mainnet_fork\conftest.py:19: KeyError
_____________ ERROR at setup of test_withdraw ____________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}

    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
>   return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
```

```
E KeyError: 'sushiswap_router'
tests\for_mainnet_fork\conftest.py:19: KeyError
_____________ ERROR at setup of test_deposit
    ↪ ____________________________

ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}

    @pytest.fixture(scope="module")
    def usdc(ERC20, network_config):
>   yield ERC20.at(network_config["usdc"])
E KeyError: 'usdc'

tests\for_mainnet_fork\conftest.py:63: KeyError
_____________ ERROR at setup of test_withdraw ________________________

ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}

    @pytest.fixture(scope="module")
    def usdc(ERC20, network_config):
>   yield ERC20.at(network_config["usdc"])
E KeyError: 'usdc'
tests\for_mainnet_fork\conftest.py:63: KeyError
__________ ERROR at setup of test_deposit ____________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
```

```
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
> return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E KeyError: 'sushiswap_router'

tests\for_mainnet_fork\conftest.py:19: KeyError
____________ ERROR at setup of test_withdraw ____________________________

UniswapV2Converter = <brownie.network.contract.ContractContainer object
    ↪ at 0x000001EF88D66520>
accounts = <brownie.network.account.Accounts object at 0
    ↪ x000001EF86307610>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def sushiswap_converter(UniswapV2Converter, accounts, network_config
        ↪ ):
> return accounts[0].deploy(UniswapV2Converter, network_config["
    ↪ sushiswap_router"])
E KeyError: 'sushiswap_router'

tests\for_mainnet_fork\conftest.py:19: KeyError
_____________ ERROR at setup of test_deposit ____________________________

WrappedERC4626YearnV2Vault = <brownie.network.contract.ContractContainer
    ↪  object at 0x000001EF884E51F0>
```

```
ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def yearn_v2_vaults(WrappedERC4626YearnV2Vault, ERC20,
        ↪ network_config):
        vaults = []
> configs = network_config["vaults_configs"]["yearn_v2"]
E KeyError: 'vaults_configs'

tests\for_mainnet_fork\conftest.py:103: KeyError
____________ ERROR at setup of test_profit _____________________________

WrappedERC4626YearnV2Vault = <brownie.network.contract.ContractContainer
    ↪  object at 0x000001EF884E51F0>
ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}

    @pytest.fixture(scope="module")
    def yearn_v2_vaults(WrappedERC4626YearnV2Vault, ERC20,
        ↪ network_config):
        vaults = []
> configs = network_config["vaults_configs"]["yearn_v2"]
E KeyError: 'vaults_configs'

tests\for_mainnet_fork\conftest.py:103: KeyError
____________ ERROR at setup of test_withdraw ___________________________

WrappedERC4626YearnV2Vault = <brownie.network.contract.ContractContainer
```

```
    ↪  object at 0x000001EF884E51F0>
ERC20 = <brownie.network.contract.ContractContainer object at 0
    ↪ x000001EF8941D520>
network_config = {'gas_limit': 'max', 'gas_buffer': 1, 'gas_price': 0, '
    ↪ max_fee': None, 'priority_fee': None, 'reverting_tx_gas_limit': '
    ↪ max', 'default_contract_owner': True, 'cmd_settings': None}
    @pytest.fixture(scope="module")
    def yearn_v2_vaults(WrappedERC4626YearnV2Vault, ERC20,
        ↪ network_config):
        vaults = []
> configs = network_config["vaults_configs"]["yearn_v2"]
E KeyError: 'vaults_configs'

tests\for_mainnet_fork\conftest.py:103: KeyError
=============== warnings summary ===================================
tests/for_local/test_deploy.py: 76 warnings
tests/for_local/erc4626_wrappers/test_totalAsstets_of_yearn.py: 50
    ↪ warnings
tests/for_local/multi-asset/test_convert_to_assets.py: 104 warnings
tests/for_local/multi-asset/test_deposit_to_apyflow.py: 122 warnings
tests/for_local/multi-asset/test_rebalance.py: 90 warnings
tests/for_local/single_asset_vault/test_deposit.py: 132 warnings
tests/for_local/single_asset_vault/test_recompute_pps_amd_harvest_fee.py
    ↪ : 60 warnings
tests/for_local/single_asset_vault/test_totalAssets.py: 72 warnings
tests/for_local/single_asset_vault/test_withdraw.py: 154 warnings
tests/for_local/test_mocks/test_converter.py: 44 warnings
tests/for_local/test_mocks/test_yearn.py: 47 warnings
  C:\Users\inas\AppData\Local\Packages\PythonSoftwareFoundation.Python
      ↪ .3.9_qbz5n2kfra8p0\LocalCache\local-packages\Python39\site-
      ↪ packages\eth_abi\codec.py:87: DeprecationWarning: abi.encode_abi
      ↪ () and abi.encode_abi_packed() are deprecated and will be
      ↪ removed in version 4.0.0 in favor of abi.encode() and abi.
      ↪ encode_packed(), respectively
```

```
    warnings.warn(
tests/for_local/test_deploy.py: 24 warnings
tests/for_local/erc4626_wrappers/test_totalAsstets_of_yearn.py: 23
    ↪ warnings
tests/for_local/multi-asset/test_convert_to_assets.py: 39 warnings
tests/for_local/multi-asset/test_deposit_to_apyflow.py: 51 warnings
tests/for_local/multi-asset/test_rebalance.py: 30 warnings
tests/for_local/single_asset_vault/test_deposit.py: 69 warnings
tests/for_local/single_asset_vault/test_recompute_pps_amd_harvest_fee.py
    ↪ : 21 warnings
tests/for_local/single_asset_vault/test_totalAssets.py: 31 warnings
tests/for_local/single_asset_vault/test_withdraw.py: 69 warnings
tests/for_local/test_mocks/test_converter.py: 25 warnings
tests/for_local/test_mocks/test_yearn.py: 23 warnings
  C:\Users\inas\AppData\Local\Packages\PythonSoftwareFoundation.Python
      ↪ .3.9_qbz5n2kfra8p0\LocalCache\local-packages\Python39\site-
      ↪ packages\eth_abi\codec.py:191: DeprecationWarning: abi.
      ↪ decode_abi() is deprecated and will be removed in version 4.0.0
      ↪ in favor of abi.decode()
    warnings.warn(

-- Docs: https://docs.pytest.org/en/stable/warnings.html
============ short test summary info ===================================
ERROR tests/for_mainnet_fork/test_asset_converter.py::test_uniswap_v2 -
    ↪ KeyError: 'cvx'
ERROR tests/for_mainnet_fork/test_asset_converter.py::test_uniswap_v3 -
    ↪ KeyError: 'weth'
ERROR tests/for_mainnet_fork/test_curve_convex_meta_vaults.py::
    ↪ test_deposit - KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_curve_convex_meta_vaults.py::
    ↪ test_compound - KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_curve_convex_meta_vaults.py::
    ↪ test_withdraw - KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_curve_convex_vaults.py::test_deposit -
```

```
    ↪  KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_curve_convex_vaults.py::test_compound
    ↪ - KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_curve_convex_vaults.py::test_withdraw
    ↪ - KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_curve_vault.py::test_deposit -
    ↪ KeyError: 'usdc'
ERROR tests/for_mainnet_fork/test_curve_vault.py::test_withdraw -
    ↪ KeyError: 'usdc'
ERROR tests/for_mainnet_fork/test_single_asset_vaults.py::test_deposit -
    ↪  KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_single_asset_vaults.py::test_withdraw
    ↪ - KeyError: 'sushiswap_router'
ERROR tests/for_mainnet_fork/test_yearn_v2_vaults.py::test_deposit -
    ↪ KeyError: 'vaults_configs'
ERROR tests/for_mainnet_fork/test_yearn_v2_vaults.py::test_profit -
    ↪ KeyError: 'vaults_configs'
ERROR tests/for_mainnet_fork/test_yearn_v2_vaults.py::test_withdraw -
    ↪ KeyError: 'vaults_configs'
========== 32 passed, 1356 warnings, 15 errors in 790.35s (0:13:10)
    ↪ ==============================
Terminating local RPC client...
```

Conclusion:

15 tests are failing, and we recommend ApyFlow to increase the branch coverage.

6 Static Analysis (Slither)

Description:

ShellBoxes expanded the coverage of the specific contract areas using automated testing methodologies. Slither, a Solidity static analysis framework, was one of the tools used. Slither was run on all-scoped contracts in both text and binary formats. This tool can be used to test mathematical relationships between Solidity instances statically and variables that allow for the detection of errors or inconsistent usage of the contracts' APIs throughout the entire codebase.

Results:

```
ICurvePool is re-used:
        - ICurvePool (contracts/mocks/CurveMock.sol#9-18)
        - ICurvePool (contracts/protocol-vaults/libraries/
          ↪ CurveMetapoolLibrary.sol#7-9)
IConvexBooster is re-used:
        - IConvexBooster (contracts/protocol-vaults/
          ↪ WrappedERC4626CurveMetapoolConvex.sol#11-17)
        - IConvexBooster (contracts/protocol-vaults/
          ↪ WrappedERC4626CurvePoolConvex.sol#11-17)
IConvexRewardVirtual is re-used:
        - IConvexRewardVirtual (contracts/protocol-vaults/
          ↪ WrappedERC4626CurveMetapoolConvex.sol#19-21)
        - IConvexRewardVirtual (contracts/protocol-vaults/
          ↪ WrappedERC4626CurvePoolConvex.sol#19-21)
IConvexReward is re-used:
        - IConvexReward (contracts/protocol-vaults/
          ↪ WrappedERC4626CurveMetapoolConvex.sol#23-35)
        - IConvexReward (contracts/protocol-vaults/
          ↪ WrappedERC4626CurvePoolConvex.sol#23-35)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #name-reused
```

```
CurveConverter.swap(address,address,uint256,address) (contracts/
    ↪ converters/CurveConverter.sol#36-45) ignores return value by
    ↪ IERC20(destination).transfer(beneficiary,result) (contracts/
    ↪ converters/CurveConverter.sol#42)
ConverterMock.swap(address,address,uint256,address) (contracts/mocks/
    ↪ ConverterMock.sol#18-28) ignores return value by IERC20(
    ↪ destination).transfer(beneficiary,result) (contracts/mocks/
    ↪ ConverterMock.sol#25)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unchecked-transfer

CurvePool.token (contracts/mocks/CurveMock.sol#27) is never initialized.
    ↪  It is used in:
        - CurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/
            ↪ CurveMock.sol#44-50)
        - CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (
            ↪ contracts/mocks/CurveMock.sol#52-60)
        - CurvePool.convertToShares(uint256) (contracts/mocks/CurveMock.
            ↪ sol#62-65)
        - CurvePool.convertToAssets(uint256) (contracts/mocks/CurveMock.
            ↪ sol#67-70)
        - CurvePool.pricePerToken() (contracts/mocks/CurveMock.sol#72-75)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #uninitialized-state-variables

Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
        -inverse = (3 * denominator) ^ 2 (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#117)
```

```
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
        -inverse *= 2 - denominator * inverse (node_modules/@openzeppelin
            ↪ /contracts/utils/math/Math.sol#121)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
        -inverse *= 2 - denominator * inverse (node_modules/@openzeppelin
            ↪ /contracts/utils/math/Math.sol#122)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
        -inverse *= 2 - denominator * inverse (node_modules/@openzeppelin
            ↪ /contracts/utils/math/Math.sol#123)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
        -inverse *= 2 - denominator * inverse (node_modules/@openzeppelin
            ↪ /contracts/utils/math/Math.sol#124)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
```

```
        -inverse *= 2 - denominator * inverse (node_modules/@openzeppelin
            ↪ /contracts/utils/math/Math.sol#125)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -denominator = denominator / twos (node_modules/@openzeppelin/
            ↪ contracts/utils/math/Math.sol#102)
        -inverse *= 2 - denominator * inverse (node_modules/@openzeppelin
            ↪ /contracts/utils/math/Math.sol#126)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) performs a multiplication
    ↪ on the result of a division:
        -prod0 = prod0 / twos (node_modules/@openzeppelin/contracts/utils
            ↪ /math/Math.sol#105)
        -result = prod0 * inverse (node_modules/@openzeppelin/contracts/
            ↪ utils/math/Math.sol#132)
ApyFlow.computeScoreDeviationInPpm(address) (contracts/ApyFlow.sol
    ↪ #199-212) performs a multiplication on the result of a division:
        -balanceAtVault = (vault.convertToAssets(shares) * (10 **
            ↪ decimals())) / (10 ** vault.decimals()) (contracts/ApyFlow
            ↪ .sol#207-208)
        -int256((1000 * balanceAtVault) / totalAssets()) - int256((1000 *
            ↪  portfolioScore) / totalPortfolioScore()) (contracts/
            ↪ ApyFlow.sol#209-211)
SingleAssetVault.recomputePricePerShareAndHarvestFee() (contracts/
    ↪ SingleAssetVault.sol#70-90) performs a multiplication on the
    ↪ result of a division:
        -newPricePerShare = (totalBalance - fee) / totalSupply() * 10 **
            ↪ decimals() (contracts/SingleAssetVault.sol#83)
WrappedERC4626CurveMetapoolConvex._convertToShares(uint256,Math.Rounding
    ↪ ) (contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.
    ↪ sol#170-181) performs a multiplication on the result of a
    ↪ division:
```

```
        -assets = (assets * (10 ** 18)) / (10 ** depositTokenDecimals) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#177)
        -metapoolLpAmount = (assets * (10 ** metapoolLpTokenDecimals)) /
            ↪ curveMetapool.getVirtualPrice() (contracts/protocol-vaults
            ↪ /WrappedERC4626CurveMetapoolConvex.sol#178-179)
WrappedERC4626CurvePool._convertToShares(uint256,Math.Rounding) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#60-69)
    ↪ performs a multiplication on the result of a division:
        -assets = (assets * (10 ** 18)) / (10 ** depositTokenDecimals) (
            ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#67)
        -shares = (assets * (10 ** decimals())) / curvePool.
            ↪ getVirtualPrice() (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePool.sol#68)
WrappedERC4626CurvePoolConvex._convertToShares(uint256,Math.Rounding) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #151-163) performs a multiplication on the result of a division:
        -assets = (assets * (10 ** 18)) / (10 ** depositTokenDecimals) (
            ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.
            ↪ sol#158)
        -_convertLpAmountToShares((assets * (10 ** lpTokenDecimals)) /
            ↪ curvePool.getVirtualPrice()) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#159-162)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #divide-before-multiply

SingleAssetVault.computeScoreDeviationInPpm(address) (contracts/
    ↪ SingleAssetVault.sol#108-115) uses a dangerous strict equality:
        - assetsInVault == 0 (contracts/SingleAssetVault.sol#113)
YearnMock.pricePerShare() (contracts/mocks/YearnMock.sol#25-28) uses a
    ↪ dangerous strict equality:
        - totalSupply() == 0 (contracts/mocks/YearnMock.sol#27)
WrappedERC4626CurveMetapoolConvex._convertMetapoolLpAmountToShares(
    ↪ uint256) (contracts/protocol-vaults/
```

```
    ↪ WrappedERC4626CurveMetapoolConvex.sol#141-155) uses a dangerous
    ↪ strict equality:
        - balance == 0 (contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#148)
WrappedERC4626CurvePoolConvex._convertLpAmountToShares(uint256) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #124-136) uses a dangerous strict equality:
        - balance == 0 (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#131)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dangerous-strict-equalities

Reentrancy in WrappedERC4626CurveMetapoolConvex._deposit(address,address
    ↪ ,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#259-272):
        External calls:
        - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#265)
        - shares = _addLiquidityAndStake(assets) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurveMetapoolConvex.sol#267)
                - metapoolLpAmount = curveMetapool.addLiquidity(amount) (
                    ↪ contracts/protocol-vaults/
                    ↪ WrappedERC4626CurveMetapoolConvex.sol#254)
                - convexBooster.deposit(convexPoolId,metapoolLpAmount,true
                    ↪ ) (contracts/protocol-vaults/
                    ↪ WrappedERC4626CurveMetapoolConvex.sol#256)
                - lpAmount = ICurveMetapoolFactoryZap3Assets(pool.
                    ↪ zapAddress).add_liquidity(pool.poolAddress,amounts
                    ↪ ,0) (contracts/protocol-vaults/libraries/
                    ↪ CurveMetapoolLibrary.sol#142-146)
                - lpAmount = ICurveMetapoolZap3Assets(pool.zapAddress).
                    ↪ add_liquidity(amounts,0) (contracts/protocol-vaults
                    ↪ /libraries/CurveMetapoolLibrary.sol#148-151)
```

```
            - lpAmount = ICurveMetapoolFactoryZap4Assets(pool.
                ↪ zapAddress).add_liquidity(pool.poolAddress,
                ↪ amounts_scope_0,0) (contracts/protocol-vaults/
                ↪ libraries/CurveMetapoolLibrary.sol#157-161)
            - lpAmount = ICurveMetapoolZap4Assets(pool.zapAddress).
                ↪ add_liquidity(amounts_scope_0,0) (contracts/
                ↪ protocol-vaults/libraries/CurveMetapoolLibrary.sol
                ↪ #163-166)
            - lpAmount = ICurveMetapoolFactoryZap5Assets(pool.
                ↪ zapAddress).add_liquidity(pool.poolAddress,
                ↪ amounts_scope_1,0) (contracts/protocol-vaults/
                ↪ libraries/CurveMetapoolLibrary.sol#172-176)
            - lpAmount = ICurveMetapoolZap5Assets(pool.zapAddress).
                ↪ add_liquidity(amounts_scope_1,0) (contracts/
                ↪ protocol-vaults/libraries/CurveMetapoolLibrary.sol
                ↪ #178-181)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#269)
                - _totalSupply += amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626CurvePoolConvex._deposit(address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#242-255):
        External calls:
        - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
            ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.
            ↪ sol#248)
        - shares = _addLiquidityAndStake(assets) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurvePoolConvex.sol#250)
                - lpAmount = curvePool.addLiquidity(amount) (contracts/
                    ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol
                    ↪ #237)
```

```
            - convexBooster.deposit(convexPoolId,lpAmount,true) (
                ↪ contracts/protocol-vaults/
                ↪ WrappedERC4626CurvePoolConvex.sol#239)
            - lpAmount = ICurvePoolAdd2AssetsReturns(pool.poolAddress)
                ↪ .add_liquidity(amounts,0) (contracts/protocol-
                ↪ vaults/libraries/CurveLibrary.sol#108-111)
            - ICurvePoolAdd2AssetsNotReturns(pool.poolAddress).
                ↪ add_liquidity(amounts,0) (contracts/protocol-vaults
                ↪ /libraries/CurveLibrary.sol#114-117)
            - lpAmount = ICurvePoolAdd3AssetsReturns(pool.poolAddress)
                ↪ .add_liquidity(amounts_scope_0,0) (contracts/
                ↪ protocol-vaults/libraries/CurveLibrary.sol#124-127)
            - ICurvePoolAdd3AssetsNotReturns(pool.poolAddress).
                ↪ add_liquidity(amounts_scope_0,0) (contracts/
                ↪ protocol-vaults/libraries/CurveLibrary.sol#130-133)
            - lpAmount = ICurvePoolAdd4AssetsReturns(pool.poolAddress)
                ↪ .add_liquidity(amounts_scope_2,0) (contracts/
                ↪ protocol-vaults/libraries/CurveLibrary.sol#140-143)
            - ICurvePoolAdd4AssetsNotReturns(pool.poolAddress).
                ↪ add_liquidity(amounts_scope_2,0) (contracts/
                ↪ protocol-vaults/libraries/CurveLibrary.sol#146-149)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#252)
                - _totalSupply += amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626CurveMetapoolConvex._withdraw(address,
    ↪ address,address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#274-289):
        External calls:
        - convexReward.withdrawAndUnwrap(metapoolLpAmount,false) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#282)
```

```
        - assets = curveMetapool.removeLiquidity(metapoolLpAmount) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#283)
        State variables written after the call(s):
        - _burn(owner,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#285)
                - _totalSupply -= amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#290)
Reentrancy in WrappedERC4626CurvePoolConvex._withdraw(address,address,
    ↪ address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#257-272):
        External calls:
        - convexReward.withdrawAndUnwrap(lpAmount,false) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#265)
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#266)
        State variables written after the call(s):
        - _burn(owner,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#268)
                - _totalSupply -= amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#290)
Reentrancy in ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol
    ↪ #136-160):
        External calls:
        - token.safeTransferFrom(msg.sender,address(this),amounts[i]) (
            ↪ contracts/ApyFlow.sol#142-146)
        - vault.deposit(amounts[i],address(this)) (contracts/ApyFlow.sol
            ↪ #155)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/ApyFlow.sol#156)
                - _totalSupply += amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol
    ↪ #30-37):
```

```
        External calls:
        - token.safeTransferFrom(msg.sender,address(this),amount) (
            ↪ contracts/mocks/YearnMock.sol#33)
        State variables written after the call(s):
        - _mint(msg.sender,shares) (contracts/mocks/YearnMock.sol#34)
                - _totalSupply += amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in SingleAssetVault.recomputePricePerShareAndHarvestFee() (
    ↪ contracts/SingleAssetVault.sol#70-90):
        External calls:
        - _withdrawAssets(feeTreasury,fee) (contracts/SingleAssetVault.
            ↪ sol#84)
                - vault.withdraw((assets * oracle.getPortfolioScore(
                    ↪ address(vault))) / totalScore,to,address(this)) (
                    ↪ contracts/SingleAssetVault.sol#215-221)
        State variables written after the call(s):
        - lastPricePerShare = newPricePerShare (contracts/
            ↪ SingleAssetVault.sol#88)
Reentrancy in UniswapV2Converter.swap(address,address,uint256,address) (
    ↪ contracts/converters/UniswapV2Converter.sol#51-80):
        External calls:
        - IERC20(source).safeIncreaseAllowance(address(router),type()(
            ↪ uint256).max) (contracts/converters/UniswapV2Converter.sol
            ↪ #69)
        State variables written after the call(s):
        - isApproved[source] = true (contracts/converters/
            ↪ UniswapV2Converter.sol#70)
Reentrancy in UniswapV3Converter.swap(address,address,uint256,address) (
    ↪ contracts/converters/UniswapV3Converter.sol#38-61):
        External calls:
        - IERC20(source).safeIncreaseAllowance(address(router),type()(
            ↪ uint256).max) (contracts/converters/UniswapV3Converter.sol
            ↪ #57)
        State variables written after the call(s):
```

```
        - isApproved[source] = true (contracts/converters/
            ↪ UniswapV3Converter.sol#58)
Reentrancy in ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol
    ↪ #162-177):
        External calls:
        - vault.withdraw(amounts[i],receiver,address(this)) (contracts/
            ↪ ApyFlow.sol#172)
        State variables written after the call(s):
        - _burn(msg.sender,shares) (contracts/ApyFlow.sol#173)
                - _totalSupply -= amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#290)
Reentrancy in YearnMock.withdraw(uint256,address) (contracts/mocks/
    ↪ YearnMock.sol#39-46):
        External calls:
        - token.safeTransfer(recipient,tokens) (contracts/mocks/YearnMock
            ↪ .sol#42)
        State variables written after the call(s):
        - _burn(msg.sender,amount) (contracts/mocks/YearnMock.sol#43)
                - _totalSupply -= amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#290)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-1

ChainlinkClient.buildChainlinkRequest(bytes32,address,bytes4).req (
    ↪ node_modules/@chainlink/contracts/src/v0.8/ChainlinkClient.sol
    ↪ #52) is a local variable never initialized
BufferChainlink.fromBytes(bytes).buf (node_modules/@chainlink/contracts/
    ↪ src/v0.8/vendor/BufferChainlink.sol#51) is a local variable never
    ↪  initialized
ChainlinkClient.buildOperatorRequest(bytes32,bytes4).req (node_modules/
    ↪ @chainlink/contracts/src/v0.8/ChainlinkClient.sol#67) is a local
    ↪ variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #uninitialized-local-variables
```

```
Chainlink.initialize(Chainlink.Request,bytes32,address,bytes4) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/Chainlink.sol#33-44)
    ↪ ignores return value by BufferChainlink.init(self.buf,
    ↪ defaultBufferSize) (node_modules/@chainlink/contracts/src/v0.8/
    ↪ Chainlink.sol#39)
Chainlink.setBuffer(Chainlink.Request,bytes) (node_modules/@chainlink/
    ↪ contracts/src/v0.8/Chainlink.sol#52-55) ignores return value by
    ↪ BufferChainlink.init(self.buf,data.length) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/Chainlink.sol#53)
Chainlink.setBuffer(Chainlink.Request,bytes) (node_modules/@chainlink/
    ↪ contracts/src/v0.8/Chainlink.sol#52-55) ignores return value by
    ↪ BufferChainlink.append(self.buf,data) (node_modules/@chainlink/
    ↪ contracts/src/v0.8/Chainlink.sol#54)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendUint8(uint8((major
    ↪ << 5) | value)) (node_modules/@chainlink/contracts/src/v0.8/
    ↪ vendor/CBORChainlink.sol#23)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendUint8(uint8((major
    ↪ << 5) | 24)) (node_modules/@chainlink/contracts/src/v0.8/vendor/
    ↪ CBORChainlink.sol#25)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendInt(value,1) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#26)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendUint8(uint8((major
    ↪ << 5) | 25)) (node_modules/@chainlink/contracts/src/v0.8/vendor/
    ↪ CBORChainlink.sol#28)
```

```
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendInt(value,2) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#29)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendUint8(uint8((major
    ↪ << 5) | 26)) (node_modules/@chainlink/contracts/src/v0.8/vendor/
    ↪ CBORChainlink.sol#31)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendInt(value,4) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#32)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendUint8(uint8((major
    ↪ << 5) | 27)) (node_modules/@chainlink/contracts/src/v0.8/vendor/
    ↪ CBORChainlink.sol#34)
CBORChainlink.encodeFixedNumeric(BufferChainlink.buffer,uint8,uint64) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#21-37) ignores return value by buf.appendInt(value,8) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#35)
CBORChainlink.encodeIndefiniteLengthType(BufferChainlink.buffer,uint8) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#39-41) ignores return value by buf.appendUint8(uint8((major
    ↪ << 5) | 31)) (node_modules/@chainlink/contracts/src/v0.8/vendor/
    ↪ CBORChainlink.sol#40)
CBORChainlink.encodeBytes(BufferChainlink.buffer,bytes) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#63-66)
    ↪ ignores return value by buf.append(value) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#65)
```

```
CBORChainlink.encodeBigNum(BufferChainlink.buffer,uint256) (node_modules
    ↪ /@chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#68-71)
    ↪ ignores return value by buf.appendUint8(uint8((MAJOR_TYPE_TAG <<
    ↪ 5) | TAG_TYPE_BIGNUM)) (node_modules/@chainlink/contracts/src/v0
    ↪ .8/vendor/CBORChainlink.sol#69)
CBORChainlink.encodeSignedBigNum(BufferChainlink.buffer,int256) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/CBORChainlink.
    ↪ sol#73-76) ignores return value by buf.appendUint8(uint8((
    ↪ MAJOR_TYPE_TAG << 5) | TAG_TYPE_NEGATIVE_BIGNUM)) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#74)
CBORChainlink.encodeString(BufferChainlink.buffer,string) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#78-81)
    ↪ ignores return value by buf.append(bytes(value)) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#80)
AccessControlEnumerable._grantRole(bytes32,address) (node_modules/
    ↪ @openzeppelin/contracts/access/AccessControlEnumerable.sol#52-55)
    ↪  ignores return value by _roleMembers[role].add(account) (
    ↪ node_modules/@openzeppelin/contracts/access/
    ↪ AccessControlEnumerable.sol#54)
AccessControlEnumerable._revokeRole(bytes32,address) (node_modules/
    ↪ @openzeppelin/contracts/access/AccessControlEnumerable.sol#60-63)
    ↪  ignores return value by _roleMembers[role].remove(account) (
    ↪ node_modules/@openzeppelin/contracts/access/
    ↪ AccessControlEnumerable.sol#62)
ApyFlow.addVault(address) (contracts/ApyFlow.sol#82-95) ignores return
    ↪ value by vaults.add(vault) (contracts/ApyFlow.sol#86)
ApyFlow.removeVault(address) (contracts/ApyFlow.sol#97-100) ignores
    ↪ return value by vaults.remove(vault) (contracts/ApyFlow.sol#99)
ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol#136-160)
    ↪ ignores return value by vault.deposit(amounts[i],address(this)) (
    ↪ contracts/ApyFlow.sol#155)
ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol#162-177)
    ↪ ignores return value by vault.withdraw(amounts[i],receiver,
    ↪ address(this)) (contracts/ApyFlow.sol#172)
```

```
ApyFlow.redeem(uint256,address) (contracts/ApyFlow.sol#179-197) ignores
    ↪ return value by vault.withdraw(amounts[i],receiver,address(this))
    ↪  (contracts/ApyFlow.sol#193)
ApyFlow.rebalance(address,address,uint256) (contracts/ApyFlow.sol
    ↪ #222-254) ignores return value by sourceVault.withdraw(assets,
    ↪ address(this),address(this)) (contracts/ApyFlow.sol#241-245)
ApyFlow.rebalance(address,address,uint256) (contracts/ApyFlow.sol
    ↪ #222-254) ignores return value by destinationVault.deposit(
    ↪ newValue,address(this)) (contracts/ApyFlow.sol#253)
SingleAssetVault.addVault(address) (contracts/SingleAssetVault.sol
    ↪ #46-51) ignores return value by vaults.add(vault) (contracts/
    ↪ SingleAssetVault.sol#49)
SingleAssetVault.removeVault(address) (contracts/SingleAssetVault.sol
    ↪ #53-59) ignores return value by vault.withdraw(vault.
    ↪ convertToAssets(vault.balanceOf(address(this))),address(this),
    ↪ address(this)) (contracts/SingleAssetVault.sol#57)
SingleAssetVault.removeVault(address) (contracts/SingleAssetVault.sol
    ↪ #53-59) ignores return value by vaults.remove(vaultAddress) (
    ↪ contracts/SingleAssetVault.sol#58)
SingleAssetVault.rebalance(address,address,uint256) (contracts/
    ↪ SingleAssetVault.sol#125-143) ignores return value by IERC4626(
    ↪ sourceVaultAddress).withdraw(assets,address(this),address(this))
    ↪ (contracts/SingleAssetVault.sol#139)
SingleAssetVault.rebalance(address,address,uint256) (contracts/
    ↪ SingleAssetVault.sol#125-143) ignores return value by IERC4626(
    ↪ destinationVaultAddress).deposit(value,address(this)) (contracts/
    ↪ SingleAssetVault.sol#142)
SingleAssetVault._deposit(address,address,uint256,uint256) (contracts/
    ↪ SingleAssetVault.sol#169-193) ignores return value by vault.
    ↪ deposit((assets * oracle.getPortfolioScore(address(vault))) /
    ↪ totalScore,address(this)) (contracts/SingleAssetVault.sol
    ↪ #180-185)
SingleAssetVault._withdrawAssets(address,uint256) (contracts/
    ↪ SingleAssetVault.sol#211-223) ignores return value by vault.
```

```
    ↪ withdraw((assets * oracle.getPortfolioScore(address(vault))) /
    ↪ totalScore,to,address(this)) (contracts/SingleAssetVault.sol
    ↪ #215-221)
CurveConverter.constructor(address,address[],int128[]) (contracts/
    ↪ converters/CurveConverter.sol#22-32) ignores return value by
    ↪ IERC20(addresses[i]).approve(curvePool,type()(uint256).max) (
    ↪ contracts/converters/CurveConverter.sol#30)
WrappedERC4626CurveMetapoolConvex._addLiquidityAndStake(uint256) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol
    ↪ #250-257) ignores return value by convexBooster.deposit(
    ↪ convexPoolId,metapoolLpAmount,true) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#256)
WrappedERC4626CurveMetapoolConvex._withdraw(address,address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#274-289) ignores return
    ↪ value by convexReward.withdrawAndUnwrap(metapoolLpAmount,false) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol
    ↪ #282)
WrappedERC4626CurveMetapoolConvex.compound() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#291-306) ignores return
    ↪ value by convexReward.getReward() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#292)
WrappedERC4626CurvePoolConvex._addLiquidityAndStake(uint256) (contracts/
    ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#233-240)
    ↪ ignores return value by convexBooster.deposit(convexPoolId,
    ↪ lpAmount,true) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#239)
WrappedERC4626CurvePoolConvex._withdraw(address,address,address,uint256,
    ↪ uint256) (contracts/protocol-vaults/WrappedERC4626CurvePoolConvex
    ↪ .sol#257-272) ignores return value by convexReward.
    ↪ withdrawAndUnwrap(lpAmount,false) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#265)
WrappedERC4626CurvePoolConvex.compound() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#274-289) ignores return value
```

```
    ↪ by convexReward.getReward() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#275)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unused-return

ENSInterface.setSubnodeOwner(bytes32,bytes32,address).owner (
    ↪ node_modules/@chainlink/contracts/src/v0.8/interfaces/
    ↪ ENSInterface.sol#20) shadows:
        - ENSInterface.owner(bytes32) (node_modules/@chainlink/contracts/
            ↪ src/v0.8/interfaces/ENSInterface.sol#29) (function)
ENSInterface.setResolver(bytes32,address).resolver (node_modules/
    ↪ @chainlink/contracts/src/v0.8/interfaces/ENSInterface.sol#23)
    ↪ shadows:
        - ENSInterface.resolver(bytes32) (node_modules/@chainlink/
            ↪ contracts/src/v0.8/interfaces/ENSInterface.sol#31) (
            ↪ function)
ENSInterface.setOwner(bytes32,address).owner (node_modules/@chainlink/
    ↪ contracts/src/v0.8/interfaces/ENSInterface.sol#25) shadows:
        - ENSInterface.owner(bytes32) (node_modules/@chainlink/contracts/
            ↪ src/v0.8/interfaces/ENSInterface.sol#29) (function)
ENSInterface.setTTL(bytes32,uint64).ttl (node_modules/@chainlink/
    ↪ contracts/src/v0.8/interfaces/ENSInterface.sol#27) shadows:
        - ENSInterface.ttl(bytes32) (node_modules/@chainlink/contracts/
            ↪ src/v0.8/interfaces/ENSInterface.sol#33) (function)
ERC20PresetFixedSupply.constructor(string,string,uint256,address).name (
    ↪ node_modules/@openzeppelin/contracts/token/ERC20/presets/
    ↪ ERC20PresetFixedSupply.sol#28) shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
ERC20PresetFixedSupply.constructor(string,string,uint256,address).symbol
    ↪  (node_modules/@openzeppelin/contracts/token/ERC20/presets/
    ↪ ERC20PresetFixedSupply.sol#29) shadows:
```

```
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
ERC20PresetMinterPauser.constructor(string,string).name (node_modules/
    ↪ @openzeppelin/contracts/token/ERC20/presets/
    ↪ ERC20PresetMinterPauser.sol#38) shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
ERC20PresetMinterPauser.constructor(string,string).symbol (node_modules/
    ↪ @openzeppelin/contracts/token/ERC20/presets/
    ↪ ERC20PresetMinterPauser.sol#38) shadows:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
SingleAssetVault.constructor(address,IERC20Metadata,string,string,
    ↪ address,uint256).name (contracts/SingleAssetVault.sol#35) shadows
    ↪ :
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
SingleAssetVault.constructor(address,IERC20Metadata,string,string,
    ↪ address,uint256).symbol (contracts/SingleAssetVault.sol#35)
    ↪ shadows:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
SingleAssetVault._withdraw(address,address,address,uint256,uint256).
    ↪ owner (contracts/SingleAssetVault.sol#198) shadows:
```

```
        - Ownable.owner() (node_modules/@openzeppelin/contracts/access/
            ↪ Ownable.sol#43-45) (function)
Token.constructor(string,string,uint256,uint8).name (contracts/mocks/
    ↪ Token.sol#13) shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
Token.constructor(string,string,uint256,uint8).symbol (contracts/mocks/
    ↪ Token.sol#13) shadows:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
YearnMock.constructor(address,string,string,uint8)._name (contracts/
    ↪ mocks/YearnMock.sol#19) shadows:
        - ERC20._name (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#42) (state variable)
YearnMock.constructor(address,string,string,uint8)._symbol (contracts/
    ↪ mocks/YearnMock.sol#19) shadows:
        - ERC20._symbol (node_modules/@openzeppelin/contracts/token/ERC20
            ↪ /ERC20.sol#43) (state variable)
WrappedERC4626CurvePool.constructor(CurveLibrary.CurvePool,string,string
    ↪ ).name (contracts/protocol-vaults/WrappedERC4626CurvePool.sol#22)
    ↪  shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
WrappedERC4626CurvePool.constructor(CurveLibrary.CurvePool,string,string
    ↪ ).symbol (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #23) shadows:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
```

```
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
WrappedERC4626CurvePoolConvex.constructor(CurveLibrary.CurvePool,address
    ↪ ,address,address,address,address,uint256,string,string).name (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol#70)
    ↪ shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
WrappedERC4626CurvePoolConvex.constructor(CurveLibrary.CurvePool,address
    ↪ ,address,address,address,address,uint256,string,string).symbol (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol#71)
    ↪ shadows:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
WrappedERC4626YearnV2Vault.constructor(IYearnV2Vault,string,string).name
    ↪  (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol#30)
    ↪ shadows:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64) (function)
        - IERC20Metadata.name() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#17) (function)
WrappedERC4626YearnV2Vault.constructor(IYearnV2Vault,string,string).
    ↪ symbol (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol
    ↪ #31) shadows:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72) (function)
        - IERC20Metadata.symbol() (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/IERC20Metadata.sol#22) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #local-variable-shadowing
```

```
PortfolioScoreOracle.constructor(address,bytes32,uint256,address,string)
    ↪ ._oracle (contracts/PortfolioScoreOracle.sol#24) lacks a zero-
    ↪ check on :
                - oracle = _oracle (contracts/PortfolioScoreOracle.sol#35)
SingleAssetVault.constructor(address,IERC20Metadata,string,string,
    ↪ address,uint256).addressForFees (contracts/SingleAssetVault.sol
    ↪ #35) lacks a zero-check on :
                - feeTreasury = addressForFees (contracts/SingleAssetVault
                    ↪ .sol#40)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-zero-address-validation

ApyFlow.totalAssets() (contracts/ApyFlow.sol#52-60) has external calls
    ↪ inside a loop: total += (vault.convertToAssets(vault.balanceOf(
    ↪ address(this))) * (10 ** decimals())) / (10 ** vault.decimals())
    ↪ (contracts/ApyFlow.sol#55-58)
ApyFlow.totalPortfolioScore() (contracts/ApyFlow.sol#110-113) has
    ↪ external calls inside a loop: total += SingleAssetVault(vaults.at
    ↪ (i)).totalPortfolioScore() (contracts/ApyFlow.sol#112)
ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol#136-160) has
    ↪ external calls inside a loop: token = IERC20(vault.asset()) (
    ↪ contracts/ApyFlow.sol#141)
ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol#136-160) has
    ↪ external calls inside a loop: vaultAssets = vault.convertToAssets
    ↪ (vault.previewDeposit(amounts[i])) (contracts/ApyFlow.sol
    ↪ #147-149)
ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol#136-160) has
    ↪ external calls inside a loop: shares = convertToShares((
    ↪ vaultAssets * (10 ** decimals())) / (10 ** vault.decimals())) (
    ↪ contracts/ApyFlow.sol#150-153)
ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol#136-160) has
    ↪ external calls inside a loop: vault.deposit(amounts[i],address(
    ↪ this)) (contracts/ApyFlow.sol#155)
```

```
ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol#162-177) has
    ↪ external calls inside a loop: shares = convertToShares((amounts[i
    ↪ ] * (10 ** decimals())) / (10 ** vault.decimals())) (contracts/
    ↪ ApyFlow.sol#167-170)
ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol#162-177) has
    ↪ external calls inside a loop: vault.withdraw(amounts[i],receiver,
    ↪ address(this)) (contracts/ApyFlow.sol#172)
ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol#162-177) has
    ↪ external calls inside a loop: Withdrawal(receiver,vault.asset(),
    ↪ amounts[i],shares) (contracts/ApyFlow.sol#174)
ApyFlow.redeem(uint256,address) (contracts/ApyFlow.sol#179-197) has
    ↪ external calls inside a loop: amounts[i] = (convertToAssets((
    ↪ shares * vault.totalPortfolioScore() / totalScore)) * (10 **
    ↪ vault.decimals())) / (10 ** 18) (contracts/ApyFlow.sol#187-192)
ApyFlow.redeem(uint256,address) (contracts/ApyFlow.sol#179-197) has
    ↪ external calls inside a loop: vault.withdraw(amounts[i],receiver,
    ↪ address(this)) (contracts/ApyFlow.sol#193)
ApyFlow.redeem(uint256,address) (contracts/ApyFlow.sol#179-197) has
    ↪ external calls inside a loop: Withdrawal(receiver,vault.asset(),
    ↪ amounts[i],shares) (contracts/ApyFlow.sol#194)
ApyFlowZap.deposit(address,uint256) (contracts/ApyFlowZap.sol#30-51) has
    ↪  external calls inside a loop: vault = SingleAssetVault(apyflow.
    ↪ getVault(i)) (contracts/ApyFlowZap.sol#39)
ApyFlowZap.deposit(address,uint256) (contracts/ApyFlowZap.sol#30-51) has
    ↪  external calls inside a loop: tokenToDeposit = vault.asset() (
    ↪ contracts/ApyFlowZap.sol#40)
ApyFlowZap.deposit(address,uint256) (contracts/ApyFlowZap.sol#30-51) has
    ↪  external calls inside a loop: amountToDeposit = vault.
    ↪ totalPortfolioScore() * value / totalPortfolioScore (contracts/
    ↪ ApyFlowZap.sol#41)
ApyFlowZap.deposit(address,uint256) (contracts/ApyFlowZap.sol#30-51) has
    ↪  external calls inside a loop: amounts[i] = assetConverter.swap(
    ↪ token,tokenToDeposit,amountToDeposit) (contracts/ApyFlowZap.sol
    ↪ #43)
```

```
ApyFlowZap.deposit(address,uint256) (contracts/ApyFlowZap.sol#30-51) has
    ↪  external calls inside a loop: IERC20(tokenToDeposit).allowance(
    ↪ address(this),address(apyflow)) < amounts[i] (contracts/
    ↪ ApyFlowZap.sol#46)
ApyFlowZap.redeem(address,uint256) (contracts/ApyFlowZap.sol#53-70) has
    ↪ external calls inside a loop: withdrawnToken = SingleAssetVault(
    ↪ apyflow.getVault(i)).asset() (contracts/ApyFlowZap.sol#58)
ApyFlowZap.redeem(address,uint256) (contracts/ApyFlowZap.sol#53-70) has
    ↪ external calls inside a loop: IERC20(withdrawnToken).allowance(
    ↪ address(this),address(assetConverter)) < amounts[i] (contracts/
    ↪ ApyFlowZap.sol#60)
ApyFlowZap.redeem(address,uint256) (contracts/ApyFlowZap.sol#53-70) has
    ↪ external calls inside a loop: assets += assetConverter.swap(
    ↪ withdrawnToken,token,amounts[i]) (contracts/ApyFlowZap.sol#63)
SingleAssetVault.totalPortfolioScore() (contracts/SingleAssetVault.sol
    ↪ #61-64) has external calls inside a loop: total += oracle.
    ↪ getPortfolioScore(vaults.at(i)) (contracts/SingleAssetVault.sol
    ↪ #63)
SingleAssetVault.totalAssets() (contracts/SingleAssetVault.sol#97-106)
    ↪ has external calls inside a loop: shares = vault.balanceOf(
    ↪ address(this)) (contracts/SingleAssetVault.sol#102)
SingleAssetVault.totalAssets() (contracts/SingleAssetVault.sol#97-106)
    ↪ has external calls inside a loop: balanceAtVault = vault.
    ↪ convertToAssets(shares) (contracts/SingleAssetVault.sol#103)
CurveConverter.constructor(address,address[],int128[]) (contracts/
    ↪ converters/CurveConverter.sol#22-32) has external calls inside a
    ↪ loop: IERC20(addresses[i]).approve(curvePool,type()(uint256).max)
    ↪  (contracts/converters/CurveConverter.sol#30)
WrappedERC4626CurveMetapoolConvex.constructor(
    ↪ WrappedERC4626CurveMetapoolConvex.ConstructorParameters) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol
    ↪ #77-135) has external calls inside a loop: i < convexReward.
    ↪ extraRewardsLength() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#110)
```

```
WrappedERC4626CurveMetapoolConvex.constructor(
    ↪ WrappedERC4626CurveMetapoolConvex.ConstructorParameters) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol
    ↪ #77-135) has external calls inside a loop: rewardToken =
    ↪ IConvexRewardVirtual(convexReward.extraRewards(i)).rewardToken()
    ↪ (contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol
    ↪ #111-113)
WrappedERC4626CurveMetapoolConvex.compound() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#291-306) has external calls
    ↪  inside a loop: balance = rewardTokens[i].balanceOf(address(this)
    ↪ ) (contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.
    ↪ sol#295)
WrappedERC4626CurveMetapoolConvex.compound() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#291-306) has external calls
    ↪  inside a loop: valueToCompound += assetConverter.swap(address(
    ↪ rewardTokens[i]),asset(),rewardTokens[i].balanceOf(address(this))
    ↪ ) (contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.
    ↪ sol#297-301)
WrappedERC4626CurvePoolConvex.constructor(CurveLibrary.CurvePool,address
    ↪ ,address,address,address,address,uint256,string,string) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #62-118) has external calls inside a loop: i < convexReward.
    ↪ extraRewardsLength() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#96)
WrappedERC4626CurvePoolConvex.constructor(CurveLibrary.CurvePool,address
    ↪ ,address,address,address,address,uint256,string,string) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #62-118) has external calls inside a loop: rewardToken =
    ↪ IConvexRewardVirtual(convexReward.extraRewards(i)).rewardToken()
    ↪ (contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #97-99)
WrappedERC4626CurvePoolConvex.compound() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#274-289) has external calls
    ↪ inside a loop: balance = rewardTokens[i].balanceOf(address(this))
```

```
    ↪  (contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #278)
WrappedERC4626CurvePoolConvex.compound() (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#274-289) has external calls
    ↪ inside a loop: valueToCompound += assetConverter.swap(address(
    ↪ rewardTokens[i]),asset(),rewardTokens[i].balanceOf(address(this))
    ↪ ) (contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol
    ↪ #280-284)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ /#calls-inside-a-loop

Reentrancy in ERC4626._deposit(address,address,uint256,uint256) (
    ↪ node_modules/@openzeppelin/contracts/token/ERC20/extensions/
    ↪ ERC4626.sol#174-191):
        External calls:
        - SafeERC20.safeTransferFrom(_asset,caller,address(this),assets)
            ↪ (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ extensions/ERC4626.sol#187)
        State variables written after the call(s):
        - _mint(receiver,shares) (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/ERC4626.sol#188)
                - _balances[account] += amount (node_modules/@openzeppelin
                    ↪ /contracts/token/ERC20/ERC20.sol#263)
        - _mint(receiver,shares) (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/extensions/ERC4626.sol#188)
                - _totalSupply += amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in SingleAssetVault._deposit(address,address,uint256,uint256)
    ↪  (contracts/SingleAssetVault.sol#169-193):
        External calls:
        - token.safeTransferFrom(caller,address(this),assets) (contracts/
            ↪ SingleAssetVault.sol#176)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/SingleAssetVault.sol#188)
```

```
            - _balances[account] += amount (node_modules/@openzeppelin
                ↪ /contracts/token/ERC20/ERC20.sol#263)
        - _mint(receiver,shares) (contracts/SingleAssetVault.sol#188)
            - _totalSupply += amount (node_modules/@openzeppelin/
                ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626CurveMetapoolConvex._deposit(address,address
    ↪ ,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#259-272):
    External calls:
    - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
        ↪ contracts/protocol-vaults/
        ↪ WrappedERC4626CurveMetapoolConvex.sol#265)
    - shares = _addLiquidityAndStake(assets) (contracts/protocol-
        ↪ vaults/WrappedERC4626CurveMetapoolConvex.sol#267)
            - metapoolLpAmount = curveMetapool.addLiquidity(amount) (
                ↪ contracts/protocol-vaults/
                ↪ WrappedERC4626CurveMetapoolConvex.sol#254)
            - convexBooster.deposit(convexPoolId,metapoolLpAmount,true
                ↪ ) (contracts/protocol-vaults/
                ↪ WrappedERC4626CurveMetapoolConvex.sol#256)
            - lpAmount = ICurveMetapoolFactoryZap3Assets(pool.
                ↪ zapAddress).add_liquidity(pool.poolAddress,amounts
                ↪ ,0) (contracts/protocol-vaults/libraries/
                ↪ CurveMetapoolLibrary.sol#142-146)
            - lpAmount = ICurveMetapoolZap3Assets(pool.zapAddress).
                ↪ add_liquidity(amounts,0) (contracts/protocol-vaults
                ↪ /libraries/CurveMetapoolLibrary.sol#148-151)
            - lpAmount = ICurveMetapoolFactoryZap4Assets(pool.
                ↪ zapAddress).add_liquidity(pool.poolAddress,
                ↪ amounts_scope_0,0) (contracts/protocol-vaults/
                ↪ libraries/CurveMetapoolLibrary.sol#157-161)
            - lpAmount = ICurveMetapoolZap4Assets(pool.zapAddress).
                ↪ add_liquidity(amounts_scope_0,0) (contracts/
                ↪ protocol-vaults/libraries/CurveMetapoolLibrary.sol
```

```
                ↪ #163-166)
            - lpAmount = ICurveMetapoolFactoryZap5Assets(pool.
                ↪ zapAddress).add_liquidity(pool.poolAddress,
                ↪ amounts_scope_1,0) (contracts/protocol-vaults/
                ↪ libraries/CurveMetapoolLibrary.sol#172-176)
            - lpAmount = ICurveMetapoolZap5Assets(pool.zapAddress).
                ↪ add_liquidity(amounts_scope_1,0) (contracts/
                ↪ protocol-vaults/libraries/CurveMetapoolLibrary.sol
                ↪ #178-181)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#269)
                - _balances[account] += amount (node_modules/@openzeppelin
                    ↪ /contracts/token/ERC20/ERC20.sol#263)
Reentrancy in WrappedERC4626CurvePool._deposit(address,address,uint256,
    ↪ uint256) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #147-161):
        External calls:
        - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
            ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#153)
        - lpAmount = curvePool.addLiquidity(assets) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurvePool.sol#155)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePool.sol#158)
                - _balances[account] += amount (node_modules/@openzeppelin
                    ↪ /contracts/token/ERC20/ERC20.sol#263)
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePool.sol#158)
                - _totalSupply += amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626CurvePoolConvex._deposit(address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#242-255):
```

```
External calls:
- IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.
    ↪ sol#248)
- shares = _addLiquidityAndStake(assets) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePoolConvex.sol#250)
        - lpAmount = curvePool.addLiquidity(amount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol
            ↪ #237)
        - convexBooster.deposit(convexPoolId,lpAmount,true) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#239)
        - lpAmount = ICurvePoolAdd2AssetsReturns(pool.poolAddress)
            ↪ .add_liquidity(amounts,0) (contracts/protocol-
            ↪ vaults/libraries/CurveLibrary.sol#108-111)
        - ICurvePoolAdd2AssetsNotReturns(pool.poolAddress).
            ↪ add_liquidity(amounts,0) (contracts/protocol-vaults
            ↪ /libraries/CurveLibrary.sol#114-117)
        - lpAmount = ICurvePoolAdd3AssetsReturns(pool.poolAddress)
            ↪ .add_liquidity(amounts_scope_0,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#124-127)
        - ICurvePoolAdd3AssetsNotReturns(pool.poolAddress).
            ↪ add_liquidity(amounts_scope_0,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#130-133)
        - lpAmount = ICurvePoolAdd4AssetsReturns(pool.poolAddress)
            ↪ .add_liquidity(amounts_scope_2,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#140-143)
        - ICurvePoolAdd4AssetsNotReturns(pool.poolAddress).
            ↪ add_liquidity(amounts_scope_2,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#146-149)
State variables written after the call(s):
- _mint(receiver,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#252)
```

```
            - _balances[account] += amount (node_modules/@openzeppelin
                ↪ /contracts/token/ERC20/ERC20.sol#263)
Reentrancy in WrappedERC4626YearnV2Vault._deposit(address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#115-128):
        External calls:
        - token.safeTransferFrom(caller,address(this),assets) (contracts/
            ↪ protocol-vaults/WrappedERC4626YearnV2Vault.sol#122)
        - yearnShares = vault.deposit(assets) (contracts/protocol-vaults/
            ↪ WrappedERC4626YearnV2Vault.sol#123)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626YearnV2Vault.sol#125)
            - _balances[account] += amount (node_modules/@openzeppelin
                ↪ /contracts/token/ERC20/ERC20.sol#263)
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626YearnV2Vault.sol#125)
            - _totalSupply += amount (node_modules/@openzeppelin/
                ↪ contracts/token/ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626CurveMetapoolConvex._withdraw(address,
    ↪ address,address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#274-289):
        External calls:
        - convexReward.withdrawAndUnwrap(metapoolLpAmount,false) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#282)
        - assets = curveMetapool.removeLiquidity(metapoolLpAmount) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#283)
        State variables written after the call(s):
        - _burn(owner,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#285)
            - _balances[account] = accountBalance - amount (
                ↪ node_modules/@openzeppelin/contracts/token/ERC20/
```

```
                ↪ ERC20.sol#288)
Reentrancy in WrappedERC4626CurvePool._withdraw(address,address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#163-177):
        External calls:
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePool.sol#171)
        State variables written after the call(s):
        - _burn(owner,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePool.sol#173)
                - _balances[account] = accountBalance - amount (
                    ↪ node_modules/@openzeppelin/contracts/token/ERC20/
                    ↪ ERC20.sol#288)
        - _burn(owner,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePool.sol#173)
                - _totalSupply -= amount (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/ERC20.sol#290)
Reentrancy in WrappedERC4626CurvePoolConvex._withdraw(address,address,
    ↪ address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#257-272):
        External calls:
        - convexReward.withdrawAndUnwrap(lpAmount,false) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#265)
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#266)
        State variables written after the call(s):
        - _burn(owner,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#268)
                - _balances[account] = accountBalance - amount (
                    ↪ node_modules/@openzeppelin/contracts/token/ERC20/
                    ↪ ERC20.sol#288)
Reentrancy in ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol
    ↪ #136-160):
        External calls:
```

```
        - token.safeTransferFrom(msg.sender,address(this),amounts[i]) (
            ↪ contracts/ApyFlow.sol#142-146)
        - vault.deposit(amounts[i],address(this)) (contracts/ApyFlow.sol
            ↪ #155)
        State variables written after the call(s):
        - _mint(receiver,shares) (contracts/ApyFlow.sol#156)
                - _balances[account] += amount (node_modules/@openzeppelin
                    ↪ /contracts/token/ERC20/ERC20.sol#263)
Reentrancy in YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol
    ↪ #30-37):
        External calls:
        - token.safeTransferFrom(msg.sender,address(this),amount) (
            ↪ contracts/mocks/YearnMock.sol#33)
        State variables written after the call(s):
        - _mint(msg.sender,shares) (contracts/mocks/YearnMock.sol#34)
                - _balances[account] += amount (node_modules/@openzeppelin
                    ↪ /contracts/token/ERC20/ERC20.sol#263)
Reentrancy in PortfolioScoreOracle.requestVaultData(address) (contracts/
    ↪ PortfolioScoreOracle.sol#41-64):
        External calls:
        - requestId = sendChainlinkRequestTo(oracle,request,fee) (
            ↪ contracts/PortfolioScoreOracle.sol#60)
                - require(bool,string)(s_link.transferAndCall(
                    ↪ oracleAddress,payment,encodedRequest),unable to
                    ↪ transferAndCall to oracle) (node_modules/@chainlink
                    ↪ /contracts/src/v0.8/ChainlinkClient.sol#173)
        State variables written after the call(s):
        - vaultForRequestId[requestId] = vaultAddress (contracts/
            ↪ PortfolioScoreOracle.sol#61)
Reentrancy in AssetConverter.updateConverter(address,address,address) (
    ↪ contracts/AssetConverter.sol#28-34):
        External calls:
        - IERC20(source).safeIncreaseAllowance(newConverter,type()(
            ↪ uint256).max) (contracts/AssetConverter.sol#31)
```

```
        State variables written after the call(s):
        - converters[source][destination] = IConverter(newConverter) (
            ↪ contracts/AssetConverter.sol#33)
Reentrancy in ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol
    ↪ #162-177):
        External calls:
        - vault.withdraw(amounts[i],receiver,address(this)) (contracts/
            ↪ ApyFlow.sol#172)
        State variables written after the call(s):
        - _burn(msg.sender,shares) (contracts/ApyFlow.sol#173)
                - _balances[account] = accountBalance - amount (
                    ↪ node_modules/@openzeppelin/contracts/token/ERC20/
                    ↪ ERC20.sol#288)
Reentrancy in YearnMock.withdraw(uint256,address) (contracts/mocks/
    ↪ YearnMock.sol#39-46):
        External calls:
        - token.safeTransfer(recipient,tokens) (contracts/mocks/YearnMock
            ↪ .sol#42)
        State variables written after the call(s):
        - _burn(msg.sender,amount) (contracts/mocks/YearnMock.sol#43)
                - _balances[account] = accountBalance - amount (
                    ↪ node_modules/@openzeppelin/contracts/token/ERC20/
                    ↪ ERC20.sol#288)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-2

Reentrancy in ERC4626._deposit(address,address,uint256,uint256) (
    ↪ node_modules/@openzeppelin/contracts/token/ERC20/extensions/
    ↪ ERC4626.sol#174-191):
        External calls:
        - SafeERC20.safeTransferFrom(_asset,caller,address(this),assets)
            ↪ (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ extensions/ERC4626.sol#187)
        Event emitted after the call(s):
```

```
        - Deposit(caller,receiver,assets,shares) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/extensions/ERC4626.sol
            ↪ #190)
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
                - _mint(receiver,shares) (node_modules/@openzeppelin/
                    ↪ contracts/token/ERC20/extensions/ERC4626.sol#188)
Reentrancy in SingleAssetVault._deposit(address,address,uint256,uint256)
    ↪  (contracts/SingleAssetVault.sol#169-193):
        External calls:
        - token.safeTransferFrom(caller,address(this),assets) (contracts/
            ↪ SingleAssetVault.sol#176)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
                - _mint(receiver,shares) (contracts/SingleAssetVault.sol
                    ↪ #188)
Reentrancy in SingleAssetVault._deposit(address,address,uint256,uint256)
    ↪  (contracts/SingleAssetVault.sol#169-193):
        External calls:
        - token.safeTransferFrom(caller,address(this),assets) (contracts/
            ↪ SingleAssetVault.sol#176)
        - recomputePricePerShareAndHarvestFee() (contracts/
            ↪ SingleAssetVault.sol#190)
                - vault.withdraw((assets * oracle.getPortfolioScore(
                    ↪ address(vault))) / totalScore,to,address(this)) (
                    ↪ contracts/SingleAssetVault.sol#215-221)
        Event emitted after the call(s):
        - Deposit(caller,receiver,assets,shares) (contracts/
            ↪ SingleAssetVault.sol#192)
        - FeeHarvested(fee,block.timestamp) (contracts/SingleAssetVault.
            ↪ sol#85)
                - recomputePricePerShareAndHarvestFee() (contracts/
                    ↪ SingleAssetVault.sol#190)
```

```
Reentrancy in WrappedERC4626CurveMetapoolConvex._deposit(address,address
    ↪ ,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#259-272):
        External calls:
        - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#265)
        - shares = _addLiquidityAndStake(assets) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurveMetapoolConvex.sol#267)
                - metapoolLpAmount = curveMetapool.addLiquidity(amount) (
                    ↪ contracts/protocol-vaults/
                    ↪ WrappedERC4626CurveMetapoolConvex.sol#254)
                - convexBooster.deposit(convexPoolId,metapoolLpAmount,true
                    ↪ ) (contracts/protocol-vaults/
                    ↪ WrappedERC4626CurveMetapoolConvex.sol#256)
                - lpAmount = ICurveMetapoolFactoryZap3Assets(pool.
                    ↪ zapAddress).add_liquidity(pool.poolAddress,amounts
                    ↪ ,0) (contracts/protocol-vaults/libraries/
                    ↪ CurveMetapoolLibrary.sol#142-146)
                - lpAmount = ICurveMetapoolZap3Assets(pool.zapAddress).
                    ↪ add_liquidity(amounts,0) (contracts/protocol-vaults
                    ↪ /libraries/CurveMetapoolLibrary.sol#148-151)
                - lpAmount = ICurveMetapoolFactoryZap4Assets(pool.
                    ↪ zapAddress).add_liquidity(pool.poolAddress,
                    ↪ amounts_scope_0,0) (contracts/protocol-vaults/
                    ↪ libraries/CurveMetapoolLibrary.sol#157-161)
                - lpAmount = ICurveMetapoolZap4Assets(pool.zapAddress).
                    ↪ add_liquidity(amounts_scope_0,0) (contracts/
                    ↪ protocol-vaults/libraries/CurveMetapoolLibrary.sol
                    ↪ #163-166)
                - lpAmount = ICurveMetapoolFactoryZap5Assets(pool.
                    ↪ zapAddress).add_liquidity(pool.poolAddress,
                    ↪ amounts_scope_1,0) (contracts/protocol-vaults/
                    ↪ libraries/CurveMetapoolLibrary.sol#172-176)
```

```
            - lpAmount = ICurveMetapoolZap5Assets(pool.zapAddress).
                ↪ add_liquidity(amounts_scope_1,0) (contracts/
                ↪ protocol-vaults/libraries/CurveMetapoolLibrary.sol
                ↪ #178-181)
        Event emitted after the call(s):
        - Deposit(caller,receiver,assets,shares) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurveMetapoolConvex.sol#271)
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
            - _mint(receiver,shares) (contracts/protocol-vaults/
                ↪ WrappedERC4626CurveMetapoolConvex.sol#269)
Reentrancy in WrappedERC4626CurvePool._deposit(address,address,uint256,
    ↪ uint256) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #147-161):
        External calls:
        - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
            ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#153)
        - lpAmount = curvePool.addLiquidity(assets) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurvePool.sol#155)
        Event emitted after the call(s):
        - Deposit(caller,receiver,assets,shares) (contracts/protocol-
            ↪ vaults/WrappedERC4626CurvePool.sol#160)
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
            - _mint(receiver,shares) (contracts/protocol-vaults/
                ↪ WrappedERC4626CurvePool.sol#158)
Reentrancy in WrappedERC4626CurvePoolConvex._deposit(address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#242-255):
        External calls:
        - IERC20(asset()).safeTransferFrom(caller,address(this),assets) (
            ↪ contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.
            ↪ sol#248)
```

```
- shares = _addLiquidityAndStake(assets) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePoolConvex.sol#250)
        - lpAmount = curvePool.addLiquidity(amount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol
            ↪ #237)
        - convexBooster.deposit(convexPoolId,lpAmount,true) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#239)
        - lpAmount = ICurvePoolAdd2AssetsReturns(pool.poolAddress)
            ↪ .add_liquidity(amounts,0) (contracts/protocol-
            ↪ vaults/libraries/CurveLibrary.sol#108-111)
        - ICurvePoolAdd2AssetsNotReturns(pool.poolAddress).
            ↪ add_liquidity(amounts,0) (contracts/protocol-vaults
            ↪ /libraries/CurveLibrary.sol#114-117)
        - lpAmount = ICurvePoolAdd3AssetsReturns(pool.poolAddress)
            ↪ .add_liquidity(amounts_scope_0,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#124-127)
        - ICurvePoolAdd3AssetsNotReturns(pool.poolAddress).
            ↪ add_liquidity(amounts_scope_0,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#130-133)
        - lpAmount = ICurvePoolAdd4AssetsReturns(pool.poolAddress)
            ↪ .add_liquidity(amounts_scope_2,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#140-143)
        - ICurvePoolAdd4AssetsNotReturns(pool.poolAddress).
            ↪ add_liquidity(amounts_scope_2,0) (contracts/
            ↪ protocol-vaults/libraries/CurveLibrary.sol#146-149)
Event emitted after the call(s):
- Deposit(caller,receiver,assets,shares) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePoolConvex.sol#254)
- Transfer(address(0),account,amount) (node_modules/@openzeppelin
    ↪ /contracts/token/ERC20/ERC20.sol#264)
        - _mint(receiver,shares) (contracts/protocol-vaults/
            ↪ WrappedERC4626CurvePoolConvex.sol#252)
```

```
Reentrancy in WrappedERC4626YearnV2Vault._deposit(address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#115-128):
        External calls:
        - token.safeTransferFrom(caller,address(this),assets) (contracts/
            ↪ protocol-vaults/WrappedERC4626YearnV2Vault.sol#122)
        - yearnShares = vault.deposit(assets) (contracts/protocol-vaults/
            ↪ WrappedERC4626YearnV2Vault.sol#123)
        Event emitted after the call(s):
        - Deposit(caller,receiver,assets,shares) (contracts/protocol-
            ↪ vaults/WrappedERC4626YearnV2Vault.sol#127)
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
                - _mint(receiver,shares) (contracts/protocol-vaults/
                    ↪ WrappedERC4626YearnV2Vault.sol#125)
Reentrancy in ERC4626._withdraw(address,address,address,uint256,uint256)
    ↪  (node_modules/@openzeppelin/contracts/token/ERC20/extensions/
    ↪ ERC4626.sol#196-217):
        External calls:
        - SafeERC20.safeTransfer(_asset,receiver,assets) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/extensions/ERC4626.sol
            ↪ #214)
        Event emitted after the call(s):
        - Withdraw(caller,receiver,owner,assets,shares) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/extensions/ERC4626.sol
            ↪ #216)
Reentrancy in SingleAssetVault._withdraw(address,address,address,uint256
    ↪ ,uint256) (contracts/SingleAssetVault.sol#195-209):
        External calls:
        - _withdrawAssets(receiver,assets) (contracts/SingleAssetVault.
            ↪ sol#204)
                - vault.withdraw((assets * oracle.getPortfolioScore(
                    ↪ address(vault))) / totalScore,to,address(this)) (
                    ↪ contracts/SingleAssetVault.sol#215-221)
```

```
        - recomputePricePerShareAndHarvestFee() (contracts/
            ↪ SingleAssetVault.sol#206)
                - vault.withdraw((assets * oracle.getPortfolioScore(
                    ↪ address(vault))) / totalScore,to,address(this)) (
                    ↪ contracts/SingleAssetVault.sol#215-221)
        Event emitted after the call(s):
        - FeeHarvested(fee,block.timestamp) (contracts/SingleAssetVault.
            ↪ sol#85)
                - recomputePricePerShareAndHarvestFee() (contracts/
                    ↪ SingleAssetVault.sol#206)
        - Withdraw(caller,receiver,owner,assets,shares) (contracts/
            ↪ SingleAssetVault.sol#208)
Reentrancy in WrappedERC4626CurveMetapoolConvex._withdraw(address,
    ↪ address,address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#274-289):
        External calls:
        - convexReward.withdrawAndUnwrap(metapoolLpAmount,false) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#282)
        - assets = curveMetapool.removeLiquidity(metapoolLpAmount) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#283)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#292)
                - _burn(owner,shares) (contracts/protocol-vaults/
                    ↪ WrappedERC4626CurveMetapoolConvex.sol#285)
Reentrancy in WrappedERC4626CurveMetapoolConvex._withdraw(address,
    ↪ address,address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#274-289):
        External calls:
        - convexReward.withdrawAndUnwrap(metapoolLpAmount,false) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#282)
```

```
        - assets = curveMetapool.removeLiquidity(metapoolLpAmount) (
            ↪ contracts/protocol-vaults/
            ↪ WrappedERC4626CurveMetapoolConvex.sol#283)
        - IERC20(asset()).safeTransfer(receiver,assets) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol#286)
        Event emitted after the call(s):
        - Withdraw(caller,receiver,owner,assets,shares) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol#288)
Reentrancy in WrappedERC4626CurvePool._withdraw(address,address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#163-177):
        External calls:
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePool.sol#171)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#292)
                - _burn(owner,shares) (contracts/protocol-vaults/
                    ↪ WrappedERC4626CurvePool.sol#173)
Reentrancy in WrappedERC4626CurvePool._withdraw(address,address,address,
    ↪ uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#163-177):
        External calls:
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePool.sol#171)
        - IERC20(asset()).safeTransfer(receiver,assets) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePool.sol#174)
        Event emitted after the call(s):
        - Withdraw(caller,receiver,owner,assets,shares) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePool.sol#176)
Reentrancy in WrappedERC4626CurvePoolConvex._withdraw(address,address,
    ↪ address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#257-272):
        External calls:
```

```
        - convexReward.withdrawAndUnwrap(lpAmount,false) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#265)
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#266)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#292)
            - _burn(owner,shares) (contracts/protocol-vaults/
                ↪ WrappedERC4626CurvePoolConvex.sol#268)
Reentrancy in WrappedERC4626CurvePoolConvex._withdraw(address,address,
    ↪ address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#257-272):
        External calls:
        - convexReward.withdrawAndUnwrap(lpAmount,false) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#265)
        - assets = curvePool.removeLiquidity(lpAmount) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#266)
        - IERC20(asset()).safeTransfer(receiver,assets) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#269)
        Event emitted after the call(s):
        - Withdraw(caller,receiver,owner,assets,shares) (contracts/
            ↪ protocol-vaults/WrappedERC4626CurvePoolConvex.sol#271)
Reentrancy in WrappedERC4626YearnV2Vault._withdraw(address,address,
    ↪ address,uint256,uint256) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#134-146):
        External calls:
        - assets = vault.withdraw(_convertSharesToYearnShares(shares),
            ↪ receiver) (contracts/protocol-vaults/
            ↪ WrappedERC4626YearnV2Vault.sol#143)
        Event emitted after the call(s):
        - Withdraw(caller,receiver,owner,assets,shares) (contracts/
            ↪ protocol-vaults/WrappedERC4626YearnV2Vault.sol#145)
Reentrancy in ApyFlow.addVault(address) (contracts/ApyFlow.sol#82-95):
        External calls:
```

```
        - token.safeIncreaseAllowance(vault,type()(uint256).max) (
            ↪ contracts/ApyFlow.sol#88)
        - token.safeIncreaseAllowance(address(assetConverter),type()(
            ↪ uint256).max) (contracts/ApyFlow.sol#89-92)
        Event emitted after the call(s):
        - NewVaultAdded(vault) (contracts/ApyFlow.sol#94)
Reentrancy in ApyFlow.deposit(uint256[],address) (contracts/ApyFlow.sol
    ↪ #136-160):
        External calls:
        - token.safeTransferFrom(msg.sender,address(this),amounts[i]) (
            ↪ contracts/ApyFlow.sol#142-146)
        - vault.deposit(amounts[i],address(this)) (contracts/ApyFlow.sol
            ↪ #155)
        Event emitted after the call(s):
        - Deposited(receiver,address(token),amounts[i],shares) (contracts
            ↪ /ApyFlow.sol#157)
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
                - _mint(receiver,shares) (contracts/ApyFlow.sol#156)
Reentrancy in YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol
    ↪ #30-37):
        External calls:
        - token.safeTransferFrom(msg.sender,address(this),amount) (
            ↪ contracts/mocks/YearnMock.sol#33)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#264)
                - _mint(msg.sender,shares) (contracts/mocks/YearnMock.sol
                    ↪ #34)
Reentrancy in SingleAssetVault.recomputePricePerShareAndHarvestFee() (
    ↪ contracts/SingleAssetVault.sol#70-90):
        External calls:
        - _withdrawAssets(feeTreasury,fee) (contracts/SingleAssetVault.
            ↪ sol#84)
```

```
            - vault.withdraw((assets * oracle.getPortfolioScore(
                ↪ address(vault))) / totalScore,to,address(this)) (
                ↪ contracts/SingleAssetVault.sol#215-221)
        Event emitted after the call(s):
        - FeeHarvested(fee,block.timestamp) (contracts/SingleAssetVault.
            ↪ sol#85)
Reentrancy in ApyFlow.redeem(uint256,address) (contracts/ApyFlow.sol
    ↪ #179-197):
        External calls:
        - vault.withdraw(amounts[i],receiver,address(this)) (contracts/
            ↪ ApyFlow.sol#193)
        Event emitted after the call(s):
        - Withdrawal(receiver,vault.asset(),amounts[i],shares) (contracts
            ↪ /ApyFlow.sol#194)
Reentrancy in PortfolioScoreOracle.requestVaultData(address) (contracts/
    ↪ PortfolioScoreOracle.sol#41-64):
        External calls:
        - requestId = sendChainlinkRequestTo(oracle,request,fee) (
            ↪ contracts/PortfolioScoreOracle.sol#60)
                - require(bool,string)(s_link.transferAndCall(
                    ↪ oracleAddress,payment,encodedRequest),unable to
                    ↪ transferAndCall to oracle) (node_modules/@chainlink
                    ↪ /contracts/src/v0.8/ChainlinkClient.sol#173)
        Event emitted after the call(s):
        - DataRequested(url,requestId) (contracts/PortfolioScoreOracle.
            ↪ sol#63)
Reentrancy in ApyFlow.withdraw(uint256[],address) (contracts/ApyFlow.sol
    ↪ #162-177):
        External calls:
        - vault.withdraw(amounts[i],receiver,address(this)) (contracts/
            ↪ ApyFlow.sol#172)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#292)
```

```
            - _burn(msg.sender,shares) (contracts/ApyFlow.sol#173)
        - Withdrawal(receiver,vault.asset(),amounts[i],shares) (contracts
            ↪ /ApyFlow.sol#174)
Reentrancy in YearnMock.withdraw(uint256,address) (contracts/mocks/
    ↪ YearnMock.sol#39-46):
        External calls:
        - token.safeTransfer(recipient,tokens) (contracts/mocks/YearnMock
            ↪ .sol#42)
        Event emitted after the call(s):
        - Transfer(account,address(0),amount) (node_modules/@openzeppelin
            ↪ /contracts/token/ERC20/ERC20.sol#292)
            - _burn(msg.sender,amount) (contracts/mocks/YearnMock.sol
                ↪ #43)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-3

BufferChainlink.init(BufferChainlink.buffer,uint256) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/BufferChainlink.sol#29-42)
    ↪ uses assembly
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#35-40)
BufferChainlink.truncate(BufferChainlink.buffer) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/BufferChainlink.sol#75-81)
    ↪ uses assembly
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#76-79)
BufferChainlink.write(BufferChainlink.buffer,uint256,bytes,uint256) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/BufferChainlink
    ↪ .sol#92-140) uses assembly
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#106-118)
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#122-124)
```

```
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#132-136)
BufferChainlink.writeUint8(BufferChainlink.buffer,uint256,uint8) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/BufferChainlink
    ↪ .sol#177-200) uses assembly
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#186-198)
BufferChainlink.write(BufferChainlink.buffer,uint256,bytes32,uint256) (
    ↪ node_modules/@chainlink/contracts/src/v0.8/vendor/BufferChainlink
    ↪ .sol#222-249) uses assembly
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#236-246)
BufferChainlink.writeInt(BufferChainlink.buffer,uint256,uint256,uint256)
    ↪  (node_modules/@chainlink/contracts/src/v0.8/vendor/
    ↪ BufferChainlink.sol#298-321) uses assembly
        - INLINE ASM (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#309-319)
Address.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/
    ↪ contracts/utils/Address.sol#201-221) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.
            ↪ sol#213-216)
Math.mulDiv(uint256,uint256,uint256) (node_modules/@openzeppelin/
    ↪ contracts/utils/math/Math.sol#55-135) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/math/
            ↪ Math.sol#66-70)
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/math/
            ↪ Math.sol#86-93)
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/math/
            ↪ Math.sol#100-109)
EnumerableSet.values(EnumerableSet.AddressSet) (node_modules/
    ↪ @openzeppelin/contracts/utils/structs/EnumerableSet.sol#282-292)
    ↪ uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/structs/
            ↪ EnumerableSet.sol#287-289)
```

```
EnumerableSet.values(EnumerableSet.UintSet) (node_modules/@openzeppelin/
    ↪ contracts/utils/structs/EnumerableSet.sol#356-366) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/structs/
            ↪ EnumerableSet.sol#361-363)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.15', '>=0.4.19', '>=0.8.0', '^0.8.0',
            ↪ '^0.8.1']
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/Chainlink.
            ↪ sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
            ↪ ChainlinkClient.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interfaces/
            ↪ ChainlinkRequestInterface.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interfaces/
            ↪ ENSInterface.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interfaces/
            ↪ LinkTokenInterface.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interfaces/
            ↪ OperatorInterface.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interfaces/
            ↪ OracleInterface.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/interfaces/
            ↪ PointerInterface.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ BufferChainlink.sol#2)
        - >=0.4.19 (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ CBORChainlink.sol#2)
        - ^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/vendor/
            ↪ ENSResolver.sol#2)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/access/
            ↪ AccessControl.sol#4)
```

```
- ^0.8.0 (node_modules/@openzeppelin/contracts/access/
  ↪ AccessControlEnumerable.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/access/
  ↪ IAccessControl.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/access/
  ↪ IAccessControlEnumerable.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/access/Ownable.sol
  ↪ #4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/interfaces/
  ↪ IERC4626.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/security/Pausable.
  ↪ sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.
  ↪ sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20
  ↪ .sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ extensions/ERC20Burnable.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ extensions/ERC20Pausable.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ extensions/ERC4626.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ extensions/IERC20Metadata.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ extensions/draft-IERC20Permit.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ presets/ERC20PresetFixedSupply.sol#3)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
  ↪ presets/ERC20PresetMinterPauser.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/
  ↪ SafeERC20.sol#4)
- ^0.8.1 (node_modules/@openzeppelin/contracts/utils/Address.sol
  ↪ #4)
```

```
- ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol
  ↪ #4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol
  ↪ #4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/utils/
  ↪ introspection/ERC165.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/utils/
  ↪ introspection/IERC165.sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/Math.
  ↪ sol#4)
- ^0.8.0 (node_modules/@openzeppelin/contracts/utils/structs/
  ↪ EnumerableSet.sol#4)
- 0.8.15 (contracts/ApyFlow.sol#3)
- >=0.8.0 (contracts/ApyFlowZap.sol#3)
- 0.8.15 (contracts/AssetConverter.sol#3)
- 0.8.15 (contracts/PortfolioScore.sol#3)
- 0.8.15 (contracts/PortfolioScoreOracle.sol#3)
- 0.8.15 (contracts/SingleAssetVault.sol#3)
- 0.8.15 (contracts/converters/CurveConverter.sol#2)
- 0.8.15 (contracts/converters/UniswapV2Converter.sol#3)
- 0.8.15 (contracts/converters/UniswapV3Converter.sol#3)
- 0.8.15 (contracts/mocks/CBridgeMock.sol#3)
- 0.8.15 (contracts/mocks/ConverterMock.sol#3)
- 0.8.15 (contracts/mocks/CurveMock.sol#3)
- 0.8.15 (contracts/mocks/MockPortfolioScore.sol#3)
- 0.8.15 (contracts/mocks/Token.sol#3)
- 0.8.15 (contracts/mocks/YearnMock.sol#3)
- 0.8.15 (contracts/protocol-vaults/
  ↪ WrappedERC4626CurveMetapoolConvex.sol#3)
- 0.8.15 (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
  ↪ #3)
- 0.8.15 (contracts/protocol-vaults/WrappedERC4626CurvePoolConvex
  ↪ .sol#3)
```

```
        - 0.8.15 (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.
            ↪ sol#3)
        - 0.8.15 (contracts/protocol-vaults/libraries/CurveLibrary.sol#3)
        - 0.8.15 (contracts/protocol-vaults/libraries/
            ↪ CurveMetapoolLibrary.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #different-pragma-directives-are-used

ERC20._mint(address,uint256) (node_modules/@openzeppelin/contracts/token
    ↪ /ERC20/ERC20.sol#257-267) has costly operations inside a loop:
        - _totalSupply += amount (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/ERC20.sol#262)
ERC20._burn(address,uint256) (node_modules/@openzeppelin/contracts/token
    ↪ /ERC20/ERC20.sol#280-295) has costly operations inside a loop:
        - _totalSupply -= amount (node_modules/@openzeppelin/contracts/
            ↪ token/ERC20/ERC20.sol#290)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #costly-operations-inside-a-loop

CurveMetapoolLibrary.calcWithdrawOneCoin(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#224-241) is never used and should be
    ↪ removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code

Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
    ↪ Chainlink.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
    ↪ ChainlinkClient.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
    ↪ interfaces/ChainlinkRequestInterface.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
    ↪ interfaces/ENSInterface.sol#2) allows old versions
```

```
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
   ↪ interfaces/LinkTokenInterface.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
   ↪ interfaces/OperatorInterface.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
   ↪ interfaces/OracleInterface.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/
   ↪ interfaces/PointerInterface.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/vendor/
   ↪ BufferChainlink.sol#2) allows old versions
Pragma version>=0.4.19 (node_modules/@chainlink/contracts/src/v0.8/
   ↪ vendor/CBORChainlink.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@chainlink/contracts/src/v0.8/vendor/
   ↪ ENSResolver.sol#2) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/
   ↪ AccessControl.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/
   ↪ AccessControlEnumerable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/
   ↪ IAccessControl.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/
   ↪ IAccessControlEnumerable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/
   ↪ Ownable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/interfaces/
   ↪ IERC4626.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/
   ↪ Pausable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ ERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ extensions/ERC20Burnable.sol#4) allows old versions
```

```
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ extensions/ERC20Pausable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ extensions/ERC4626.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ extensions/IERC20Metadata.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ extensions/draft-IERC20Permit.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ presets/ERC20PresetFixedSupply.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ presets/ERC20PresetMinterPauser.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/
   ↪ utils/SafeERC20.sol#4) allows old versions
Pragma version^0.8.1 (node_modules/@openzeppelin/contracts/utils/Address
   ↪ .sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context
   ↪ .sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings
   ↪ .sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/
   ↪ introspection/ERC165.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/
   ↪ introspection/IERC165.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/math/
   ↪ Math.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/structs
   ↪ /EnumerableSet.sol#4) allows old versions
Pragma version0.8.15 (contracts/ApyFlow.sol#3) necessitates a version
   ↪ too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version>=0.8.0 (contracts/ApyFlowZap.sol#3) allows old versions
Pragma version0.8.15 (contracts/AssetConverter.sol#3) necessitates a
   ↪ version too recent to be trusted. Consider deploying with
```

```
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/PortfolioScore.sol#3) necessitates a
   ↪ version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/PortfolioScoreOracle.sol#3) necessitates
   ↪  a version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/SingleAssetVault.sol#3) necessitates a
   ↪ version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/converters/CurveConverter.sol#2)
   ↪ necessitates a version too recent to be trusted. Consider
   ↪ deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/converters/UniswapV2Converter.sol#3)
   ↪ necessitates a version too recent to be trusted. Consider
   ↪ deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/converters/UniswapV3Converter.sol#3)
   ↪ necessitates a version too recent to be trusted. Consider
   ↪ deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/mocks/CBridgeMock.sol#3) necessitates a
   ↪ version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/mocks/ConverterMock.sol#3) necessitates
   ↪ a version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/mocks/CurveMock.sol#3) necessitates a
   ↪ version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/mocks/MockPortfolioScore.sol#3)
   ↪ necessitates a version too recent to be trusted. Consider
   ↪ deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/mocks/Token.sol#3) necessitates a
   ↪ version too recent to be trusted. Consider deploying with
   ↪ 0.6.12/0.7.6/0.8.7
```

```
Pragma version0.8.15 (contracts/mocks/YearnMock.sol#3) necessitates a
    ↪ version too recent to be trusted. Consider deploying with
    ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/protocol-vaults/
    ↪ WrappedERC4626CurveMetapoolConvex.sol#3) necessitates a version
    ↪ too recent to be trusted. Consider deploying with
    ↪ 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/protocol-vaults/WrappedERC4626CurvePool.
    ↪ sol#3) necessitates a version too recent to be trusted. Consider
    ↪ deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#3) necessitates a version too
    ↪ recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#3) necessitates a version too
    ↪ recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/protocol-vaults/libraries/CurveLibrary.
    ↪ sol#3) necessitates a version too recent to be trusted. Consider
    ↪ deploying with 0.6.12/0.7.6/0.8.7
Pragma version0.8.15 (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#3) necessitates a version too recent to
    ↪ be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
solc-0.8.15 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/
    ↪ @openzeppelin/contracts/utils/Address.sol#60-65):
        - (success) = recipient.call{value: amount}() (node_modules/
            ↪ @openzeppelin/contracts/utils/Address.sol#63)
Low level call in Address.functionCallWithValue(address,bytes,uint256,
    ↪ string) (node_modules/@openzeppelin/contracts/utils/Address.sol
    ↪ #128-139):
```

```
        - (success,returndata) = target.call{value: value}(data) (
            ↪ node_modules/@openzeppelin/contracts/utils/Address.sol
            ↪ #137)
Low level call in Address.functionStaticCall(address,bytes,string) (
    ↪ node_modules/@openzeppelin/contracts/utils/Address.sol#157-166):
        - (success,returndata) = target.staticcall(data) (node_modules/
            ↪ @openzeppelin/contracts/utils/Address.sol#164)
Low level call in Address.functionDelegateCall(address,bytes,string) (
    ↪ node_modules/@openzeppelin/contracts/utils/Address.sol#184-193):
        - (success,returndata) = target.delegatecall(data) (node_modules/
            ↪ @openzeppelin/contracts/utils/Address.sol#191)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #low-level-calls

CurveConverter (contracts/converters/CurveConverter.sol#16-47) should
    ↪ inherit from IConverter (contracts/AssetConverter.sol#11-14)
UniswapV2Converter (contracts/converters/UniswapV2Converter.sol#35-81)
    ↪ should inherit from IConverter (contracts/AssetConverter.sol
    ↪ #11-14)
UniswapV3Converter (contracts/converters/UniswapV3Converter.sol#27-62)
    ↪ should inherit from IConverter (contracts/AssetConverter.sol
    ↪ #11-14)
ConverterMock (contracts/mocks/ConverterMock.sol#9-30) should inherit
    ↪ from IConverter (contracts/AssetConverter.sol#11-14)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-inheritance

Constant Chainlink.defaultBufferSize (node_modules/@chainlink/contracts/
    ↪ src/v0.8/Chainlink.sol#12) is not in UPPER_CASE_WITH_UNDERSCORES
Variable ChainlinkClient.s_ens (node_modules/@chainlink/contracts/src/v0
    ↪ .8/ChainlinkClient.sol#29) is not in mixedCase
Variable ChainlinkClient.s_ensNode (node_modules/@chainlink/contracts/
    ↪ src/v0.8/ChainlinkClient.sol#30) is not in mixedCase
```

```
Variable ChainlinkClient.s_link (node_modules/@chainlink/contracts/src/
    ↪ v0.8/ChainlinkClient.sol#31) is not in mixedCase
Variable ChainlinkClient.s_oracle (node_modules/@chainlink/contracts/src
    ↪ /v0.8/ChainlinkClient.sol#32) is not in mixedCase
Variable ChainlinkClient.s_requestCount (node_modules/@chainlink/
    ↪ contracts/src/v0.8/ChainlinkClient.sol#33) is not in mixedCase
Variable ChainlinkClient.s_pendingRequests (node_modules/@chainlink/
    ↪ contracts/src/v0.8/ChainlinkClient.sol#34) is not in mixedCase
Struct BufferChainlink.buffer (node_modules/@chainlink/contracts/src/v0
    ↪ .8/vendor/BufferChainlink.sol#18-21) is not in CapWords
Function IERC20Permit.DOMAIN_SEPARATOR() (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/extensions/draft-IERC20Permit.sol#59) is
    ↪ not in mixedCase
Parameter ICurve.exchange(int128,int128,uint256,uint256).min_dy (
    ↪ contracts/converters/CurveConverter.sol#10) is not in mixedCase
Function ICurve.get_dy(int128,int128,uint256) (contracts/converters/
    ↪ CurveConverter.sol#12) is not in mixedCase
Function IUniswapV2Router.WETH() (contracts/converters/
    ↪ UniswapV2Converter.sol#12) is not in mixedCase
Variable UniswapV2Converter.WETH (contracts/converters/
    ↪ UniswapV2Converter.sol#41) is not in mixedCase
Function ICurvePool.get_virtual_price() (contracts/mocks/CurveMock.sol
    ↪ #11) is not in mixedCase
Function ICurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/
    ↪ CurveMock.sol#13) is not in mixedCase
Parameter ICurvePool.add_liquidity(uint256[3],uint256).min_mint_amount (
    ↪ contracts/mocks/CurveMock.sol#13) is not in mixedCase
Function ICurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (
    ↪ contracts/mocks/CurveMock.sol#15) is not in mixedCase
Parameter ICurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ token_amount (contracts/mocks/CurveMock.sol#15) is not in
    ↪ mixedCase
Parameter ICurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ min_amount (contracts/mocks/CurveMock.sol#15) is not in mixedCase
```

```
Function ICurvePool.calc_token_amount(uint256[3],bool) (contracts/mocks/
    ↪ CurveMock.sol#17) is not in mixedCase
Parameter ICurvePool.calc_token_amount(uint256[3],bool).is_deposit (
    ↪ contracts/mocks/CurveMock.sol#17) is not in mixedCase
Function CurvePool.get_virtual_price() (contracts/mocks/CurveMock.sol
    ↪ #34-37) is not in mixedCase
Function CurvePool.calc_token_amount(uint256[3],bool) (contracts/mocks/
    ↪ CurveMock.sol#39-42) is not in mixedCase
Parameter CurvePool.calc_token_amount(uint256[3],bool).is_deposit (
    ↪ contracts/mocks/CurveMock.sol#39) is not in mixedCase
Function CurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/
    ↪ CurveMock.sol#44-50) is not in mixedCase
Parameter CurvePool.add_liquidity(uint256[3],uint256).min_mint_amount (
    ↪ contracts/mocks/CurveMock.sol#44) is not in mixedCase
Function CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (
    ↪ contracts/mocks/CurveMock.sol#52-60) is not in mixedCase
Parameter CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ token_amount (contracts/mocks/CurveMock.sol#52) is not in
    ↪ mixedCase
Parameter CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ min_amount (contracts/mocks/CurveMock.sol#52) is not in mixedCase
Variable CurvePool.lp_token (contracts/mocks/CurveMock.sol#25) is not in
    ↪  mixedCase
Function ICurvePoolView.calc_withdraw_one_coin(uint256,int128) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#8-11) is not
    ↪  in mixedCase
Parameter ICurvePoolView.calc_withdraw_one_coin(uint256,int128).
    ↪ token_amount (contracts/protocol-vaults/libraries/CurveLibrary.
    ↪ sol#8) is not in mixedCase
Function ICurvePoolView.get_virtual_price() (contracts/protocol-vaults/
    ↪ libraries/CurveLibrary.sol#13) is not in mixedCase
Function ICurvePoolRemoveReturns.remove_liquidity_one_coin(uint256,
    ↪ int128,uint256) (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#17-21) is not in mixedCase
```

```
Parameter ICurvePoolRemoveReturns.remove_liquidity_one_coin(uint256,
    ↪ int128,uint256).token_amount (contracts/protocol-vaults/libraries
    ↪ /CurveLibrary.sol#18) is not in mixedCase
Parameter ICurvePoolRemoveReturns.remove_liquidity_one_coin(uint256,
    ↪ int128,uint256).min_amount (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#20) is not in mixedCase
Function ICurvePoolRemoveNotReturns.remove_liquidity_one_coin(uint256,
    ↪ int128,uint256) (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#25-29) is not in mixedCase
Parameter ICurvePoolRemoveNotReturns.remove_liquidity_one_coin(uint256,
    ↪ int128,uint256).token_amount (contracts/protocol-vaults/libraries
    ↪ /CurveLibrary.sol#26) is not in mixedCase
Parameter ICurvePoolRemoveNotReturns.remove_liquidity_one_coin(uint256,
    ↪ int128,uint256).min_amount (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#28) is not in mixedCase
Function ICurvePoolCalc2Assets.calc_token_amount(uint256[2],bool) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#33-36) is
    ↪ not in mixedCase
Parameter ICurvePoolCalc2Assets.calc_token_amount(uint256[2],bool).
    ↪ is_deposit (contracts/protocol-vaults/libraries/CurveLibrary.sol
    ↪ #33) is not in mixedCase
Function ICurvePoolAdd2AssetsReturns.add_liquidity(uint256[2],uint256) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#40-42) is
    ↪ not in mixedCase
Parameter ICurvePoolAdd2AssetsReturns.add_liquidity(uint256[2],uint256).
    ↪ min_mint_amount (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#40) is not in mixedCase
Function ICurvePoolAdd2AssetsNotReturns.add_liquidity(uint256[2],uint256
    ↪ ) (contracts/protocol-vaults/libraries/CurveLibrary.sol#46-47) is
    ↪  not in mixedCase
Parameter ICurvePoolAdd2AssetsNotReturns.add_liquidity(uint256[2],
    ↪ uint256).min_mint_amount (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#46) is not in mixedCase
```

```
Function ICurvePoolCalc3Assets.calc_token_amount(uint256[3],bool) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#51-54) is
    ↪ not in mixedCase
Parameter ICurvePoolCalc3Assets.calc_token_amount(uint256[3],bool).
    ↪ is_deposit (contracts/protocol-vaults/libraries/CurveLibrary.sol
    ↪ #51) is not in mixedCase
Function ICurvePoolAdd3AssetsReturns.add_liquidity(uint256[3],uint256) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#58-60) is
    ↪ not in mixedCase
Parameter ICurvePoolAdd3AssetsReturns.add_liquidity(uint256[3],uint256).
    ↪ min_mint_amount (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#58) is not in mixedCase
Function ICurvePoolAdd3AssetsNotReturns.add_liquidity(uint256[3],uint256
    ↪ ) (contracts/protocol-vaults/libraries/CurveLibrary.sol#64-65) is
    ↪  not in mixedCase
Parameter ICurvePoolAdd3AssetsNotReturns.add_liquidity(uint256[3],
    ↪ uint256).min_mint_amount (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#64) is not in mixedCase
Function ICurvePoolCalc4Assets.calc_token_amount(uint256[4],bool) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#69-72) is
    ↪ not in mixedCase
Parameter ICurvePoolCalc4Assets.calc_token_amount(uint256[4],bool).
    ↪ is_deposit (contracts/protocol-vaults/libraries/CurveLibrary.sol
    ↪ #69) is not in mixedCase
Function ICurvePoolAdd4AssetsReturns.add_liquidity(uint256[4],uint256) (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#76-78) is
    ↪ not in mixedCase
Parameter ICurvePoolAdd4AssetsReturns.add_liquidity(uint256[4],uint256).
    ↪ min_mint_amount (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#76) is not in mixedCase
Function ICurvePoolAdd4AssetsNotReturns.add_liquidity(uint256[4],uint256
    ↪ ) (contracts/protocol-vaults/libraries/CurveLibrary.sol#82-83) is
    ↪  not in mixedCase
```

```
Parameter ICurvePoolAdd4AssetsNotReturns.add_liquidity(uint256[4],
    ↪ uint256).min_mint_amount (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#82) is not in mixedCase
Function ICurveMetapoolFactoryZap.calc_withdraw_one_coin(address,uint256
    ↪ ,int128) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#12-16) is not in mixedCase
Parameter ICurveMetapoolFactoryZap.calc_withdraw_one_coin(address,
    ↪ uint256,int128).token_amount (contracts/protocol-vaults/libraries
    ↪ /CurveMetapoolLibrary.sol#14) is not in mixedCase
Function ICurveMetapoolFactoryZap.remove_liquidity_one_coin(address,
    ↪ uint256,int128,uint256) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#18-23) is not in mixedCase
Parameter ICurveMetapoolFactoryZap.remove_liquidity_one_coin(address,
    ↪ uint256,int128,uint256).token_amount (contracts/protocol-vaults/
    ↪ libraries/CurveMetapoolLibrary.sol#20) is not in mixedCase
Parameter ICurveMetapoolFactoryZap.remove_liquidity_one_coin(address,
    ↪ uint256,int128,uint256).min_amount (contracts/protocol-vaults/
    ↪ libraries/CurveMetapoolLibrary.sol#22) is not in mixedCase
Function ICurveMetapoolZap.calc_withdraw_one_coin(uint256,int128) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #27-30) is not in mixedCase
Parameter ICurveMetapoolZap.calc_withdraw_one_coin(uint256,int128).
    ↪ token_amount (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#28) is not in mixedCase
Function ICurveMetapoolZap.remove_liquidity_one_coin(uint256,int128,
    ↪ uint256) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#32-36) is not in mixedCase
Parameter ICurveMetapoolZap.remove_liquidity_one_coin(uint256,int128,
    ↪ uint256).token_amount (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#33) is not in mixedCase
Parameter ICurveMetapoolZap.remove_liquidity_one_coin(uint256,int128,
    ↪ uint256).min_amount (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#35) is not in mixedCase
```

```
Function ICurveMetapoolFactoryZap3Assets.calc_token_amount(address,
    ↪ uint256[3],bool) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#40-44) is not in mixedCase
Parameter ICurveMetapoolFactoryZap3Assets.calc_token_amount(address,
    ↪ uint256[3],bool).is_deposit (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#43) is not in mixedCase
Function ICurveMetapoolFactoryZap3Assets.add_liquidity(address,uint256
    ↪ [3],uint256) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#46-50) is not in mixedCase
Parameter ICurveMetapoolFactoryZap3Assets.add_liquidity(address,uint256
    ↪ [3],uint256).min_mint_amount (contracts/protocol-vaults/libraries
    ↪ /CurveMetapoolLibrary.sol#49) is not in mixedCase
Function ICurveMetapoolZap3Assets.calc_token_amount(uint256[3],bool) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #54-57) is not in mixedCase
Parameter ICurveMetapoolZap3Assets.calc_token_amount(uint256[3],bool).
    ↪ is_deposit (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#56) is not in mixedCase
Function ICurveMetapoolZap3Assets.add_liquidity(uint256[3],uint256) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #59-62) is not in mixedCase
Parameter ICurveMetapoolZap3Assets.add_liquidity(uint256[3],uint256).
    ↪ min_mint_amount (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#61) is not in mixedCase
Function ICurveMetapoolFactoryZap4Assets.calc_token_amount(address,
    ↪ uint256[4],bool) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#66-70) is not in mixedCase
Parameter ICurveMetapoolFactoryZap4Assets.calc_token_amount(address,
    ↪ uint256[4],bool).is_deposit (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#69) is not in mixedCase
Function ICurveMetapoolFactoryZap4Assets.add_liquidity(address,uint256
    ↪ [4],uint256) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#72-76) is not in mixedCase
```

```
Parameter ICurveMetapoolFactoryZap4Assets.add_liquidity(address,uint256
    ↪ [4],uint256).min_mint_amount (contracts/protocol-vaults/libraries
    ↪ /CurveMetapoolLibrary.sol#75) is not in mixedCase
Function ICurveMetapoolZap4Assets.calc_token_amount(uint256[4],bool) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #80-83) is not in mixedCase
Parameter ICurveMetapoolZap4Assets.calc_token_amount(uint256[4],bool).
    ↪ is_deposit (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#82) is not in mixedCase
Function ICurveMetapoolZap4Assets.add_liquidity(uint256[4],uint256) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #85-88) is not in mixedCase
Parameter ICurveMetapoolZap4Assets.add_liquidity(uint256[4],uint256).
    ↪ min_mint_amount (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#87) is not in mixedCase
Function ICurveMetapoolFactoryZap5Assets.calc_token_amount(address,
    ↪ uint256[5],bool) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#92-96) is not in mixedCase
Parameter ICurveMetapoolFactoryZap5Assets.calc_token_amount(address,
    ↪ uint256[5],bool).is_deposit (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#95) is not in mixedCase
Function ICurveMetapoolFactoryZap5Assets.add_liquidity(address,uint256
    ↪ [5],uint256) (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#98-102) is not in mixedCase
Parameter ICurveMetapoolFactoryZap5Assets.add_liquidity(address,uint256
    ↪ [5],uint256).min_mint_amount (contracts/protocol-vaults/libraries
    ↪ /CurveMetapoolLibrary.sol#101) is not in mixedCase
Function ICurveMetapoolZap5Assets.calc_token_amount(uint256[5],bool) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #106-109) is not in mixedCase
Parameter ICurveMetapoolZap5Assets.calc_token_amount(uint256[5],bool).
    ↪ is_deposit (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#108) is not in mixedCase
```

```
Function ICurveMetapoolZap5Assets.add_liquidity(uint256[5],uint256) (
    ↪ contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol
    ↪ #111-114) is not in mixedCase
Parameter ICurveMetapoolZap5Assets.add_liquidity(uint256[5],uint256).
    ↪ min_mint_amount (contracts/protocol-vaults/libraries/
    ↪ CurveMetapoolLibrary.sol#113) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

Variable ApyFlow.rebalance(address,address,uint256).scoreDeviation1 (
    ↪ contracts/ApyFlow.sol#230) is too similar to ApyFlow.rebalance(
    ↪ address,address,uint256).scoreDeviation2 (contracts/ApyFlow.sol
    ↪ #231)
Variable SingleAssetVault.rebalance(address,address,uint256).
    ↪ scoreDeviation1 (contracts/SingleAssetVault.sol#132) is too
    ↪ similar to SingleAssetVault.rebalance(address,address,uint256).
    ↪ scoreDeviation2 (contracts/SingleAssetVault.sol#133)
Variable CurveLibrary.calcTokenAmount(CurveLibrary.CurvePool,uint256,
    ↪ bool).amounts_scope_0 (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#168) is too similar to CurveLibrary.
    ↪ calcTokenAmount(CurveLibrary.CurvePool,uint256,bool).
    ↪ amounts_scope_1 (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#175)
Variable CurveLibrary.calcTokenAmount(CurveLibrary.CurvePool,uint256,
    ↪ bool).amounts_scope_0 (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#168) is too similar to CurveLibrary.addLiquidity
    ↪ (CurveLibrary.CurvePool,uint256).amounts_scope_2 (contracts/
    ↪ protocol-vaults/libraries/CurveLibrary.sol#137)
Variable CurveLibrary.addLiquidity(CurveLibrary.CurvePool,uint256).
    ↪ amounts_scope_0 (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#121) is too similar to CurveLibrary.calcTokenAmount(
    ↪ CurveLibrary.CurvePool,uint256,bool).amounts_scope_1 (contracts/
    ↪ protocol-vaults/libraries/CurveLibrary.sol#175)
```

```
Variable CurveLibrary.calcTokenAmount(CurveLibrary.CurvePool,uint256,
    ↪ bool).amounts_scope_1 (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#175) is too similar to CurveLibrary.addLiquidity
    ↪ (CurveLibrary.CurvePool,uint256).amounts_scope_2 (contracts/
    ↪ protocol-vaults/libraries/CurveLibrary.sol#137)
Variable CurveLibrary.addLiquidity(CurveLibrary.CurvePool,uint256).
    ↪ amounts_scope_0 (contracts/protocol-vaults/libraries/CurveLibrary
    ↪ .sol#121) is too similar to CurveLibrary.addLiquidity(
    ↪ CurveLibrary.CurvePool,uint256).amounts_scope_2 (contracts/
    ↪ protocol-vaults/libraries/CurveLibrary.sol#137)
Variable CurveLibrary.addLiquidity(CurveLibrary.CurvePool,uint256).
    ↪ previousBalanace_scope_1 (contracts/protocol-vaults/libraries/
    ↪ CurveLibrary.sol#129) is too similar to CurveLibrary.addLiquidity
    ↪ (CurveLibrary.CurvePool,uint256).previousBalanace_scope_3 (
    ↪ contracts/protocol-vaults/libraries/CurveLibrary.sol#145)
Variable CurveMetapoolLibrary.calcTokenAmount(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256,bool).amounts_scope_0 (contracts/protocol-
    ↪ vaults/libraries/CurveMetapoolLibrary.sol#202) is too similar to
    ↪ CurveMetapoolLibrary.addLiquidity(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256).amounts_scope_1 (contracts/protocol-vaults
    ↪ /libraries/CurveMetapoolLibrary.sol#169)
Variable CurveMetapoolLibrary.calcTokenAmount(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256,bool).amounts_scope_0 (contracts/protocol-
    ↪ vaults/libraries/CurveMetapoolLibrary.sol#202) is too similar to
    ↪ CurveMetapoolLibrary.calcTokenAmount(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256,bool).amounts_scope_1 (contracts/protocol-
    ↪ vaults/libraries/CurveMetapoolLibrary.sol#212)
Variable CurveMetapoolLibrary.addLiquidity(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256).amounts_scope_0 (contracts/protocol-vaults
    ↪ /libraries/CurveMetapoolLibrary.sol#154) is too similar to
    ↪ CurveMetapoolLibrary.calcTokenAmount(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256,bool).amounts_scope_1 (contracts/protocol-
    ↪ vaults/libraries/CurveMetapoolLibrary.sol#212)
```

```
Variable CurveMetapoolLibrary.addLiquidity(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256).amounts_scope_0 (contracts/protocol-vaults
    ↪ /libraries/CurveMetapoolLibrary.sol#154) is too similar to
    ↪ CurveMetapoolLibrary.addLiquidity(CurveMetapoolLibrary.
    ↪ CurveMetapool,uint256).amounts_scope_1 (contracts/protocol-vaults
    ↪ /libraries/CurveMetapoolLibrary.sol#169)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #variable-names-are-too-similar

CBORChainlink.encodeInt(BufferChainlink.buffer,int256) (node_modules/
    ↪ @chainlink/contracts/src/v0.8/vendor/CBORChainlink.sol#51-61)
    ↪ uses literals with too many digits:
        - value < - 0x10000000000000000 (node_modules/@chainlink/
          ↪ contracts/src/v0.8/vendor/CBORChainlink.sol#52)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #too-many-digits

ChainlinkClient.LINK_DIVISIBILITY (node_modules/@chainlink/contracts/src
    ↪ /v0.8/ChainlinkClient.sol#20) is never used in
    ↪ PortfolioScoreOracle (contracts/PortfolioScoreOracle.sol#10-112)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unused-state-variable

WrappedERC4626CurveMetapoolConvex.poolAssetIndex (contracts/protocol-
    ↪ vaults/WrappedERC4626CurveMetapoolConvex.sol#50) should be
    ↪ constant
WrappedERC4626CurveMetapoolConvex.poolAssetsCount (contracts/protocol-
    ↪ vaults/WrappedERC4626CurveMetapoolConvex.sol#51) should be
    ↪ constant
WrappedERC4626CurvePoolConvex.poolAssetIndex (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePoolConvex.sol#49) should be constant
WrappedERC4626CurvePoolConvex.poolAssetsCount (contracts/protocol-vaults
    ↪ /WrappedERC4626CurvePoolConvex.sol#50) should be constant
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #state-variables-that-could-be-declared-constant

addr(bytes32) should be declared external:
        - ENSResolver.addr(bytes32) (node_modules/@chainlink/contracts/
            ↪ src/v0.8/vendor/ENSResolver.sol#5)
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (node_modules/
            ↪ @openzeppelin/contracts/access/AccessControl.sol#144-146)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (node_modules/
            ↪ @openzeppelin/contracts/access/AccessControl.sol#159-161)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (node_modules/
            ↪ @openzeppelin/contracts/access/AccessControl.sol#179-183)
getRoleMember(bytes32,uint256) should be declared external:
        - AccessControlEnumerable.getRoleMember(bytes32,uint256) (
            ↪ node_modules/@openzeppelin/contracts/access/
            ↪ AccessControlEnumerable.sol#37-39)
getRoleMemberCount(bytes32) should be declared external:
        - AccessControlEnumerable.getRoleMemberCount(bytes32) (
            ↪ node_modules/@openzeppelin/contracts/access/
            ↪ AccessControlEnumerable.sol#45-47)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (node_modules/@openzeppelin/
            ↪ contracts/access/Ownable.sol#61-63)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (node_modules/@openzeppelin/
            ↪ contracts/access/Ownable.sol#69-72)
name() should be declared external:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/
            ↪ ERC20.sol#62-64)
symbol() should be declared external:
```

```
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/
            ↪ ERC20/ERC20.sol#70-72)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/ERC20.sol#113-117)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/ERC20.sol#136-140)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/ERC20.sol#158-167)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/ERC20.sol#181-185)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/ERC20.sol#201-210)
burn(uint256) should be declared external:
        - ERC20Burnable.burn(uint256) (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/extensions/ERC20Burnable.sol#20-22)
burnFrom(address,uint256) should be declared external:
        - ERC20Burnable.burnFrom(address,uint256) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/extensions/
            ↪ ERC20Burnable.sol#35-38)
convertToShares(uint256) should be declared external:
        - ERC4626.convertToShares(uint256) (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/extensions/ERC4626.sol#49-51)
deposit(uint256,address) should be declared external:
        - ERC4626.deposit(uint256,address) (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/extensions/ERC4626.sol#99-106)
mint(uint256,address) should be declared external:
        - ERC4626.mint(uint256,address) (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/extensions/ERC4626.sol#109-116)
withdraw(uint256,address,address) should be declared external:
```

```
        - ERC4626.withdraw(uint256,address,address) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/extensions/ERC4626.sol
            ↪ #119-130)
redeem(uint256,address,address) should be declared external:
        - ERC4626.redeem(uint256,address,address) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/extensions/ERC4626.sol
            ↪ #133-144)
mint(address,uint256) should be declared external:
        - ERC20PresetMinterPauser.mint(address,uint256) (node_modules/
            ↪ @openzeppelin/contracts/token/ERC20/presets/
            ↪ ERC20PresetMinterPauser.sol#54-57)
pause() should be declared external:
        - ERC20PresetMinterPauser.pause() (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/presets/ERC20PresetMinterPauser.sol
            ↪ #68-71)
unpause() should be declared external:
        - ERC20PresetMinterPauser.unpause() (node_modules/@openzeppelin/
            ↪ contracts/token/ERC20/presets/ERC20PresetMinterPauser.sol
            ↪ #82-85)
pricePerToken() should be declared external:
        - ApyFlow.pricePerToken() (contracts/ApyFlow.sol#78-80)
requestVaultData(address) should be declared external:
        - PortfolioScoreOracle.requestVaultData(address) (contracts/
            ↪ PortfolioScoreOracle.sol#41-64)
fulfill(bytes32,uint256,uint256,uint256,uint256,uint256,uint256,uint256,
    ↪ uint256) should be declared external:
        - PortfolioScoreOracle.fulfill(bytes32,uint256,uint256,uint256,
            ↪ uint256,uint256,uint256,uint256,uint256) (contracts/
            ↪ PortfolioScoreOracle.sol#66-87)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #public-function-that-could-be-declared-external
. analyzed (90 contracts with 78 detectors), 388 result(s) found
```

Conclusion:

Most of the vulnerabilities found by the analysis have already been addressed by the smart contract code review.

7 Conclusion

In this audit, we examined the design and implementation of ApyFlow V2 contract and discovered several issues of varying severity. ApyFlow team addressed 13 issues raised in the initial report and implemented the necessary fixes, while classifying the rest as a risk with low-probability of occurrence. Shellboxes' auditors advised ApyFlow Team to maintain a high level of vigilance and to keep those findings in mind in order to avoid any future complications.

8 Scope Files

8.1 Audit

Files	MD5 Hash
contracts/ApyFlow.sol	a2a9e7727a65f5ce6dadcf37006baaf6
contracts/ApyFlowZap.sol	750af3e0f074e6d9c0bbba91a7f1a28d
contracts/AssetConverter.sol	30d6b46c1f620d2c680c9f9c800aecc3
contracts/PortfolioScore.sol	9d2c7478e40c04baa96483b7619b868d
contracts/PortfolioScoreOracle.sol	38bf005695be7047e49dae51cccbb70c
contracts/SingleAssetVault.sol	7a6dc859068df5a21bed0cfaa6498ce6
contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol	b4d9f7917e37e766fad63c4a510231ab
contracts/protocol-vaults/WrappedERC4626CurvePool.sol	5347a2ade0f84fc9ca329775ff65ae06
contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol	bad87265604ab9cf649f745d2c6f6dbd
contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol	79fa051ee7862b9b89d0d052f2a0a206
contracts/protocol-vaults/libraries/CurveLibrary.sol	ba84c1151a9b53b91797025c6503277b
contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol	dc458c541b04470b341495b841429603
contracts/mocks/CBridgeMock.sol	eebec9f5928814d8aa96450a0497f1f2
contracts/mocks/ConverterMock.sol	5dc413b07389924f12d18c01b2b6793c

contracts/mocks/CurveMock.sol	40a488dfc71458111a5b561241eebbe8
contracts/mocks/MockPortfolioScore.sol	eec31bdf19dc0c631dec49bd460ad812
contracts/mocks/Token.sol	65215f1d1537c12e71fb1477785d0c01
contracts/mocks/YearnMock.sol	97b22a6410d35f000e1e8cbc27dcfbd2
contracts/converters/CurveConverter.sol	e5fc75c0792f069cf161202a7d756a7a
contracts/converters/UniswapV2Converter.sol	3d46d8cb56653c20f40acd3a6ff2deb7
contracts/converters/UniswapV3Converter.sol	fd605d527af9d0e456396a5644d020b5

8.2 Re-Audit

Files	MD5 Hash
contracts/ApyFlow.sol	c7556a3f77019298a03b0aaccf58515a
contracts/ApyFlowZap.sol	5605d02807b37917943e8cdc986e23b7
contracts/AssetConverter.sol	410b054134bfebb444ced24284a07afa
contracts/PortfolioScore.sol	13cf7fac093afa79b71fd64725b98b63
contracts/PricePerTokenMixin.sol	ded2b08b65638e8c1d5a73973380df9d
contracts/SingleAssetVault.sol	8deda8f4173c29edb8ff6480e09bca8c
contracts/protocol-vaults/WrappedERC4626CurveMetapoolConvex.sol	9ed1c1624bac03d74aed66d4b4d69f11
contracts/protocol-vaults/WrappedERC4626CurvePool.sol	62dff389ac564dc258ca4a218139f7b3
contracts/protocol-vaults/WrappedERC4626CurvePoolConvex.sol	1743108d892fe6c10276a098c958bf6d

contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol	1a64177967f53c7c1b46543c84be779a
contracts/protocol-vaults/libraries/CurveLibrary.sol	ba84c1151a9b53b91797025c6503277b
contracts/protocol-vaults/libraries/CurveMetapoolLibrary.sol	dc458c541b04470b341495b841429603
contracts/mocks/CBridgeMock.sol	eebec9f5928814d8aa96450a0497f1f2
contracts/mocks/ConverterMock.sol	d8b82f15028ab2ebc1501ef9fcac202a
contracts/mocks/CurveMock.sol	40a488dfc71458111a5b561241eebbe8
contracts/mocks/Token.sol	65215f1d1537c12e71fb1477785d0c01
contracts/mocks/YearnMock.sol	5d6ababe7b7ac9a847dc3babbe10cc84
contracts/converters/CurveConverter.sol	73b633734a04f4028596e818701a3fda
contracts/converters/UniswapV2Converter.sol	2810baa0e237c5cd4e57b6fdf96a05b8

9 Disclaimer

Shellboxes reports should not be construed as "endorsements" or "disapprovals" of particular teams or projects. These reports do not reflect the economics or value of any "product" or "asset" produced by any team or project that engages Shellboxes to do a security evaluation, nor should they be regarded as such. Shellboxes Reports do not provide any warranty or guarantee regarding the absolute bug-free nature of the examined technology, nor do they provide any indication of the technology's proprietors, business model, business or legal compliance. Shellboxes Reports should not be used in any way to decide whether to invest in or take part in a certain project. These reports don't offer any kind of investing advice and shouldn't be used that way. Shellboxes Reports are the result of a thorough auditing process designed to assist our clients in improving the quality of their code while lowering the significant risk posed by blockchain technology. According to Shellboxes, each business and person is in charge of their own due diligence and ongoing security. Shellboxes does not guarantee the security or functionality of the technology we agree to research; instead, our purpose is to assist in limiting the attack vectors and the high degree of variation associated with using new and evolving technologies.

For a Contract Audit, contact us at contact@shellboxes.com