ApyFlow

Smart Contract Security Audit

Prepared by ShellBoxes

Sep 7th, 2022 - Sep 15th, 2022

Shellboxes.com

contact@shellboxes.com

Document Properties

Client	ApyFlow
Version	1
Classification	Public

Scope

The ApyFlow Contract in the ApyFlow Repository

Repo	Commit Hash
https://gitlab.com/apyflowcom/apyflow-dlt	2976075d1e595d13f08365df531be528d70cfc8c

Files	MD5 Hash
contracts/ApyFlow.sol	a2c31a2b91e300f6fa3e20cad3051bb9
contracts/AssetConverter.sol	c62fb9bb36bd72c9b19cde4b0e3014da
contracts/PortfolioScore.sol	dc5a5c819e2cce7cf732faf5c9641ea9
contracts/PortfolioScoreOracle.sol	25438e3f87999e792297dca603dd89ec
contracts/SingleAssetVault.sol	f3f1cc6423aaba73b841e70f7efbf6f5

Re-Audit

Repo	Commit Hash
https://gitlab.com/apyflowcom/apyflow-dlt	bdfc71064758e56a53f43165b9a9f118bc691251

Re-Audit Files

Files	MD5 Hash
contracts/ApyFlow.sol	8882790b1a960efd6f844ad4e5e8b7c9
contracts/AssetConverter.sol	06595e86b72015a2e304032b516c403a
contracts/PortfolioScore.sol	9d2c7478e40c04baa96483b7619b868d
contracts/PortfolioScoreOracle.sol	38bf005695be7047e49dae51cccbb70c
contracts/SingleAssetVault.sol	7c404e4bcc5f5690964d63e1744e8bcc
contracts/protocol-vaults/WrappedERC4626CurvePool.sol	5347a2ade0f84fc9ca329775ff65ae06
contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol	79fa051ee7862b9b89d0d052f2a0a206
contracts/converters/CurveConverter.sol	221efaa568344fd26584c1ce77600e27

Contacts

COMPANY	EMAIL
ShellBoxes	contact@shellboxes.com

Contents

1 Introduction

ApyFlow engaged ShellBoxes to conduct a security assessment on the ApyFlow beginning on Sep 7th, 2022 and ending Sep 15th, 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				
	High	Critical	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Low
		High	Medium	Low

Likelihood

2 Findings Overview

2.1 Summary

The following is a synopsis of our conclusions from our analysis of the ApyFlow 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, 2 high-severity, 3 medium-severity, 18 low-severity vulnerabilities.

Vulnerabilities	Severity	Status
B.1. Users Can Deposit Any Token	CRITICAL	Fixed
B.2. pricePerToken Can Have An Incorrect Value	CRITICAL	Fixed
B.3. Comission Fees Are Excluded In The rebalance deposit withdraw Function	CRITICAL	Acknowledged
B.4. Loss Precision	CRITICAL	Mitigated
G.1. Race Condition In get_virtual_price	HIGH	Mitigated
H.1. Race Condition In pricePerShare	HIGH	Mitigated
B.5. Missing Transfer Verification	MEDIUM	Fixed
C.1. Missing Transfer Verification	MEDIUM	Fixed
F.1. Owner Can Add Duplicate Vaults	MEDIUM	Fixed
A.1. Missing Address Verification	LOW	Fixed
A.2. Renounce Ownership	LOW	Acknowledged
A.3. Floating Pragma	LOW	Fixed
B.6. Missing Value Verification	LOW	Mitigated
B.7. Missing Address Verification	LOW	Fixed

B.8. Approve Race	LOW	Fixed
B.9. Renounce Ownership	LOW	Acknowledged
C.2. Renounce Ownership	LOW	Acknowledged
C.3. Floating Pragma	LOW	Fixed
D.1. Floating Pragma	LOW	Fixed
E.1. Floating Pragma	LOW	Fixed
F.2. Missing Address Verification	LOW	Fixed
F.3. Renounce Ownership	LOW	Acknowledged
F.4. Floating Pragma	LOW	Fixed
G.2. Missing Address Verification	LOW	Fixed
G.3. Floating Pragma	LOW	Fixed
H.2. Missing Address Verification	LOW	Fixed
H.3. Floating Pragma	LOW	Fixed

3 Finding Details

A CurveConverter.sol

A.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, some of the contract's functionality may become inaccessible.

Code:

Listing 1: CurveConverter.sol

```
17      constructor (address curvePool) Ownable()
18      {
19          pool = ICurve(curvePool);
20      }
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

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

Status - Fixed

The team has resolved the issue by adding the necessary verifications.

Listing 2: CurveConverter.sol

```
24    require(curvePool != address(0), "Null address provided");
```

A.2 Renounce Ownership [LOW]

Description:

Typically, the contract's owner is the account that deploys the contract. As a result, the owner can perform certain privileged activities. The renounceOwnership function is used in smart contracts to renounce ownership. However, if the contract's ownership has never been transferred before renouncing it, it will never have an Owner, which may result in a denial of service.

Code:

Listing 3: CurveConverter.sol

```
13  contract Converter is Ownable
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It is advised that the Owner cannot call renounceOwnership without first transferring ownership to a different address. Additionally, if a multi-signature wallet is utilized, executing the renounceOwnership method will require two or more users to sign the transaction. Alternatively, the Renounce Ownership functionality can be disabled by overriding it.

Status - Acknowledged

The team has acknowledged the risk knowing that they will use a multisig wallet for these actions.

A.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 and flags that were used during the testing process. 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 4: CurveConverter.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

B ApyFlow.sol

B.1 Users Can Deposit Any Token [CRITICAL]

Description:

In the deposit function, the user can transfer any ERC20 token to the contract by inserting the address of the token, however when discussing with the team only a few tokens are supported in this process. These supported assets are stored in the tokens array.

Code:

Listing 5: ApyFlow.sol

```
77  function deposit(address token, uint value) external
78  {
79   IERC20(token).transferFrom(msg.sender, address(this), value);
80   uint tokensNumber = value / pricePerToken;
81   vaultsForToken[token].vault.deposit(value, address(this));
82   _mint(msg.sender, tokensNumber);
83   emit Deposited(msg.sender, token, value, toRekensNumber);
84  }
```

Risk Level:

Likelihood – 5
Impact - 5

Recommendation:

Consider verifying if vaultsForToken[token].vault is different from address(0), if it's the case revert the transaction.

Status - Fixed

The team has resolved the issue by verifying if vaultsForToken[token].vault exists.

Code:

Listing 6: ApyFlow.sol

```
94  function deposit(address token, uint value) external returns (uint256
        ↪ shares)
95  {
96   require(!vaultsForToken[token].isDeprecated, "vault is depreacted now
        ↪ ");
97   require(address(vaultsForToken[token].vault) != address(0), "token can
        ↪  be deposited");
```

B.2 pricePerToken Can Have An Incorrect Value [CRITICAL]

Description:

In the deposit,withdraw functions, the tokensNumber is calculated using the pricePerToken which has a default value of 1. This variable is only changed by calling the recomputePricePerToken, the issue here is that there is no guarantee that this function will be called by the user or ApyFlow causing the pricePerToken to not change.

Code:

Listing 7: ApyFlow.sol

```
77 function deposit(address token, uint value) external
78  {
79   IERC20(token).transferFrom(msg.sender, address(this), value);
80   uint tokensNumber = value / pricePerToken;
```

Listing 8: ApyFlow.sol

```
87  function withdraw(address token, uint value, address recipient)
        ↪ external
88  {
89   uint tokensNumber = value / pricePerToken;
90   _burn(msg.sender, tokensNumber);
```

Risk Level:

Likelihood – 4
Impact - 5

Recommendation:

Consider calling the recomputePricePerToken in withdraw and deposit functions before assigning the tokensNumber.

Status - Fixed

The team has resolved the issue by calculating pricePerToken directly using the convertToShares function.

Code:

Listing 9: ApyFlow.sol

```
94 function deposit(address token, uint value) external returns (uint256
       ↪ shares)
95  {
96   require(!vaultsForToken[token].isDeprecated, "vault is depreacted now
       ↪ ");
97   require(address(vaultsForToken[token].vault) != address(0), "token can
       ↪  be deposited");
98   IERC20(token).safeTransferFrom(msg.sender, address(this), value);
99   uint tokensNumber = convertToShares(value) * 10 ** decimals() / 10 **
       ↪ IERC20Metadata(token).decimals();
```

Listing 10: ApyFlow.sol

```
108 function deposit(address token, uint value) external returns (uint256
       ↪ shares)
109 function withdraw(address token, uint value, address recipient) external
       ↪  returns (uint256 assets)
110  {
```

```
111   uint tokensNumber = convertToShares(value) * 10 ** decimals() / 10 **
          ↪ IERC20Metadata(token).decimals();
112   _burn(msg.sender, tokensNumber);
113   IERC4626 vault = vaultsForToken[token].vault;
114   assets = vault.withdraw(value, recipient, address(this));
```

B.3 Comission Fees Are Excluded In The rebalance deposit withdraw Function [CRITICAL]

Description:

In the white paper it's mentioned that ApyFlow charges a 20% profit fee. Those fees will be charged once a month or if there are deposit,withdrawal,rebalance events. In the contracts there is no such logic of commissions fees.

Code:

Listing 11: ApyFlow.sol

```
77  function deposit(address token, uint value) external
78  {
79   IERC20(token).transferFrom(msg.sender, address(this), value);
80   uint tokensNumber = value / pricePerToken;
81   vaultsForToken[token].vault.deposit(value, address(this));
82   _mint(msg.sender, tokensNumber);
83   emit Deposited(msg.sender, token, value, toRekensNumber);
84  }
```

Listing 12: ApyFlow.sol

```
87  function withdraw(address token, uint value, address recipient)
        ↪ external
88  {
89   uint tokensNumber = value / pricePerToken;
90   _burn(msg.sender, tokensNumber);
```

```
91    IERC4626 vault = vaultsForToken[token].vault;
92    vault.withdraw(value, recipient, address(this));
93    emit Withdrawal(msg.sender, token, value, tokensNumber);
94   }
```

Listing 13: ApyFlow.sol

```
106  function rebalance(address sourceToken, address destinationToken,
        ↪ uint256 assets) external
107  {
108   int256 scoreDeviation1 = computeScoreDeviationInPpm(sourceToken);
109   int256 scoreDeviation2 = computeScoreDeviationInPpm(destinationToken);
110   require(scoreDeviation1 > scoreDeviation2);
111   uint256 value = vaultsForToken[sourceToken].vault.withdraw(assets,
        ↪ address(this), address(this));
112   uint256 newValue = assetConverter.swap(sourceToken, destinationToken,
        ↪ value);
113   vaultsForToken[destinationToken].vault.deposit(newValue, address(this)
        ↪ );
114   require(computeScoreDeviationInPpm(sourceToken) >
        ↪ computeScoreDeviationInPpm(destinationToken));
115  }
```

Risk Level:

Likelihood – 5
Impact - 5

Recommendation:

Consider adding in the functions, a function beforeActions that will take the 20% commission fees.

Status - Acknowledged

The team has acknowledged the risk knowing that they are charging fees on the SingleAssetVault.

B.4 Loss Precision [CRITICAL]

Description:

In the deposit function, the tokensNumber are calculated by dividing the value by the pricePerToken and minting the amount to the caller.
If the value is less than pricePerToken the tokensNumber will be 0 due to loss of precision, and therefore the user will transfer a number of value tokens to the contracts without minting any tokens, causing the user to lose his tokens.
- Same issue in the computeScoreDeviationInPpm (L103).

Code:

Listing 14: ApyFlow.sol

```
77  function deposit(address token, uint value) external
78  {
79   IERC20(token).transferFrom(msg.sender, address(this), value);
80   uint tokensNumber = value / pricePerToken;
81   vaultsForToken[token].vault.deposit(value, address(this));
82   _mint(msg.sender, tokensNumber);
```

Code:

Listing 15: ApyFlow.sol

```
97  function computeScoreDeviationInPpm(address token) public view returns
        ↪ (int256)
98  {
99   VaultInfo storage vaultInfo = vaultsForToken[token];
```

```
100    uint256 portfolioScore = vaultInfo.vault.totalPortfolioScore();
101    uint shares = vaultInfo.vault.balanceOf(address(this));
102    uint balanceAtVault = vaultInfo.vault.convertToAssets(shares);
103    return int256(1000 * portfolioScore / totalPortfolioScore) - int256
          ↪ (1000 * balanceAtVault / totalAssets);
```

Risk Level:

Likelihood – 4
Impact - 5

Recommendation:

It's recommended to verify first if value is greater than pricePerToken else revert.

Status - Mitigated

The team has mitigated the risk by using multiplication operations before division.

B.5 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 successful. 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. However, it is mostly missed by developers when they carry out checks in effect, the transaction would always succeed, even if the token transfer did not.

Code:

Listing 16: ApyFlow.sol

```
79  function deposit(address token, uint value) external
80  {
81   IERC20(token).transferFrom(msg.sender, address(this), value);
```

Risk Level:

Likelihood – 2
Impact - 5

Recommendation:

Use the safeTransferFrom function from the safeERC20 Implementation, or put the transfer call inside an assert or require to verify that the transfer has passed successfully.

Status - Fixed

The team has resolved the issue by using the safeTransferFrom function from OpenZeppelin library.

B.6 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.
- The APY variable should be less than 100

Code:

Listing 17: ApyFlow.sol

```
49   vaultsForToken[token] = VaultInfo(SingleAssetVault(vault), apy, false)
        ↪ ;
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It's recommended to verify the values provided in the arguments. The concerns can be resolved by utilizing a require statement.

Status - Mitigated

The team has mitigated the risk by removing the apy variable.

B.7 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, some of the contract's functionality may become inaccessible.

Code:

Listing 18: ApyFlow.sol

```
40  constructor (address converter) ERC20("ApyFlow", "APYFLW")
41  {
42   assetConverter = AssetConverter(converter);
43  }
```

Listing 19: ApyFlow.sol

```
45 function addNewVault(address token, address vault, uint apy) external
       ↪ onlyOwner
46  {
47   require(address(vaultsForToken[token].vault) == address(0));
```

```
48    vaultsForToken[token] = VaultInfo(SingleAssetVault(vault), apy, false)
          ↪ ;
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

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

Status - Fixed

The team has resolved the issue by adding the necessary verifications.

Code:

Listing 20: ApyFlow.sol

```
40  constructor (address _converter, address _oracle) ERC20("ApyFlow", "
        ↪ APYFLW")
41  {
42   require(_converter != address(0), "Zero address provided");
43   require(_oracle != address(0), "Zero address provided");
44   assetConverter = AssetConverter(_converter);
```

Listing 21: ApyFlow.sol

```
76 unction addNewVault(address token, address vault) external onlyOwner
77  {
78   require(address(vaultsForToken[token].vault) == address(0), "Check
        ↪ that vault for that token has not initialized yet");
79   require(vault != address(0), "Zero address provided");
80   vaultsForToken[token] = VaultInfo(SingleAssetVault(vault), false);
```

B.8 Approve Race [LOW]

Description:

The standard ERC20 implementation contains a widely known racing condition in its approve function, wherein a spender can witness the token owner broadcast a transaction altering their approval and quickly sign and broadcast a transaction using transferFrom to move the current approved amount from the owner's balance to the spender. If the spender's transaction is validated before the owner's, the spender will be able to get both approval amounts of both transactions.

Code:

Listing 22: ApyFlow.sol

```
12 contract PistonTokenController is Initializable, ERC20Upgradeable,
       ↪ OwnableUpgradeable {
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It is recommended to use the increaseAllowance() and decreaseAllowance() function to override the approval amount instead of the approve() function.

Status - Fixed

The team has resolved the issue by using the SafeERC20 which supports the increaseAllowance() and decreaseAllowance() functions.

B.9 Renounce Ownership [LOW]

Description:

Typically, the contract's owner is the account that deploys the contract. As a result, the owner can perform certain privileged activities. The renounceOwnership function is used in smart contracts to renounce ownership. However, if the contract's ownership has never been transferred before renouncing it, it will never have an Owner, which may result in a denial of service.

Code:

Listing 23: ApyFlow.sol

```
12 contract ApyFlow is ERC20, Ownable
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It is advised that the Owner cannot call renounceOwnership without first transferring ownership to a different address. Additionally, if a multi-signature wallet is utilized, executing the renounceOwnership method will require two or more users to sign the transaction. Alternatively, the Renounce Ownership functionality can be disabled by overriding it.

Status - Acknowledged

The team has acknowledged the risk knowing that they will use a multisig wallet for these actions.

C AssetConverter.sol

C.1 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 successful. 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. However, it is mostly missed by developers when they carry out checks in effect, the transaction would always succeed, even if the token transfer did not.

Code:

Listing 24: AssetConverter.sol

```
32  function swap(address source, address destination, uint256 value)
        ↪ external returns (uint256)
33  {
34   IERC20(source).transferFrom(msg.sender, address(this), value);
35   return converters[source][destination].swap(source, destination, value
        ↪ , msg.sender);
36  }
```

Risk Level:

Likelihood – 2
Impact - 5

Recommendation:

Use the safeTransferFrom function from the safeERC20 Implementation, or put the transfer call inside an assert or require to verify that the transfer has passed successfully.

Status - Fixed

The team has resolved the issue by using the safeTransferFrom function from OpenZeppelin library.

C.2 Renounce Ownership [LOW]

Description:

Typically, the contract's owner is the account that deploys the contract. As a result, the owner can perform certain privileged activities. The renounceOwnership function is used in smart contracts to renounce ownership. However, if the contract's ownership has never been transferred before renouncing it, it will never have an Owner, which may result in a denial of service.

Code:

Listing 25: AssetConverter.sol

```
5 contract AssetConverter is Ownable
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It is advised that the Owner cannot call renounceOwnership without first transferring ownership to a different address. Additionally, if a multi-signature wallet is utilized, executing the renounceOwnership method will require two or more users to sign the transaction. Alternatively, the Renounce Ownership functionality can be disabled by overriding it.

Status - Acknowledged

The team has acknowledged the risk knowing that they will use a multisig wallet for these actions.

C.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 and flags that were used during the testing process. 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 26: AssetConverter.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

D PortofolioScore.sol

D.1 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 and flags that were used during the testing process. 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 27: PortofolioScore.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

E PortfolioScoreOracle.sol

E.1 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 and flags that were used during the testing process. 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 28: PortfolioScoreOracle.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

F SingleAssetVault.sol

F.1 Owner Can Add Duplicate Vaults [MEDIUM]

Description:

The owner can add new values using the addNewVault function, the issue here is that there is no verification on the vault if it already exists, causing the duplication of vaults and counting the totalPortofolioScore two times.

Code:

Listing 29: SingleAssetVault.sol

```
36  function addNewVault(IERC4626 vault) external onlyOwner
37  {
38   uint256 portfolioScore = oracle.getPortfolioScore(address(vault));
39   vaults.push(VaultInfo(vault, false, 0));
40   IERC20(asset()).approve(address(vault), type(uint256).max);
41   totalPortfolioScore += portfolioScore;
42  }
```

Risk Level:

Likelihood – 2
Impact - 4

Recommendation:

It is recommended to add a function isVaultExisting and loop over the vaults array to verify if the vault doesn't exist.

Status - Fixed

The team has resolved the issue by adding the isVaultExisting mapping, and check for each new vault creation if the vault already exist.

Code:

Listing 30: SingleAssetVault.sol

```
50  function addNewVault(IERC4626 vault) external onlyOwner
51  {
52   require(!isVaultExisting[address(vault)], "prevent double addition of
        ↪ one vault");

54   isVaultExisting[address(vault)] = true;
```

F.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, some of the contract's functionality may become inaccessible.

Code:

Listing 31: SingleAssetVault.sol

```
30  constructor (address portfolioScore, IERC20Metadata asset_, string
        ↪ memory name, string memory symbol) ERC4626(asset_) ERC20(name,
        ↪ symbol)
31  {
32   assetsInSingleAssetVault = 1; // kostyl
33   oracle = PortfolioScore(portfolioScore);
34  }
```

Risk Level:

Likelihood – 1

Impact - 3

Recommendation:

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

Status - Fixed

The team has resolved the issue by adding the necessary verifications.

Code:

Listing 32: SingleAssetVault.sol

```
39  constructor (address portfolioScore, IERC20Metadata asset_, string
        ↪ memory name, string memory symbol, address addressForFees,
        ↪ uint256 _fee) ERC4626(asset_) ERC20(name, symbol)
40  {
41   require(portfolioScore != address(0), "Zero address provided");
42   oracle = PortfolioScore(portfolioScore);
```

F.3 Renounce Ownership [LOW]

Description:

Typically, the contract's owner is the account that deploys the contract. As a result, the owner can perform certain privileged activities. The renounceOwnership function is used in smart contracts to renounce ownership. However, if the contract's ownership has never been transferred before renouncing it, it will never have an Owner, which may result in a denial of service.

Code:

Listing 33: SingleAssetVault.sol

```
11  contract SingleAssetVault is ERC4626, Ownable
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It is advised that the Owner cannot call renounceOwnership without first transferring ownership to a different address. Additionally, if a multi-signature wallet is utilized, executing the renounceOwnership method will require two or more users to sign the transaction. Alternatively, the Renounce Ownership functionality can be disabled by overriding it.

Status - Acknowledged

The team has acknowledged the risk knowing that they will use a multisig wallet for these actions.

F.4 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 and flags that were used during the testing process. 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 34: SingleAssetVault.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

G WrappedERC4626CurvePool.sol

G.1 Race Condition In get_virtual_price [HIGH]

Description:

The _convertToShares and _convertToAssets functions use the get_virtual_price from CurvePool contract. The issue here is that there is a big probability to have a desynchronization between what the user is seeing using the view function in CurvePool and the real amount calculated after the transaction is executed, causing the user to make a decision based on some wrong inputs.

Code:

Listing 35: WrappedERC4626CurvePool.sol

```
37  function _convertToShares(uint256 assets, Math.Rounding rounding)
        ↪ internal view virtual override returns (uint256 shares)
38  {
39   return assets / curvePool.get_virtual_price();
40  }
```

Listing 36: WrappedERC4626CurvePool.sol

```
43  function _convertToAssets(uint256 shares, Math.Rounding rounding)
        ↪ internal view virtual override returns (uint256 assets)
```

```
44  {
45   return shares * curvePool.get_virtual_price();
46  }
```

Risk Level:

Likelihood – 3
Impact - 5

Recommendation:

We recommend adding the value of get_virtual_price from the front in the parameter of functions that call _convertToShares and _convertToAssets and verify if the two values match (the value in the front and the curvePool.get_virtual_price() if it's not the case revert the transaction).

Status - Mitigated

The team has mitigated the risk by using static calls to get real precision.

G.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, some of the contract's functionality may become inaccessible.

Code:

Listing 37: WrappedERC4626CurvePool.sol

```
19  constructor (address curve, address crv, address asset, string memory
        ↪ name, string memory symbol) ERC4626(IERC20Metadata(crv)) ERC20(
        ↪ name, symbol)
20  {
```

```
21    curvePool = ICurvePool(curve);
22    lpToken = IERC20(crv);
23    token = IERC20(asset);
24    token.approve(curve, type(uint256).max);
25    lpToken.approve(curve, type(uint256).max);
26  }
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

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

Status - Fixed

The team has resolved the issue by adding the necessary verifications.

Code:

Listing 38: WrappedERC4626CurvePool.sol

```
18    constructor(
19        CurveLibrary.CurvePool memory _curvePool,
20        string memory name,
21        string memory symbol
22     ) ERC4626(IERC20Metadata(_curvePool.depositToken)) ERC20(name,
          ↪ symbol) {
23        curvePool = _curvePool;
24        require(_curvePool.poolAddress != address(0), "Zero address
              ↪ provided");
25        require(address(_curvePool.LPToken) != address(0), "Zero address
              ↪ provided");
```

```
26          require(address(_curvePool.depositToken) != address(0), "Zero
                ↪ address provided");

28          lpTokenDecimals = curvePool.LPToken.decimals();
29          depositTokenDecimals = _curvePool.depositToken.decimals();
```

G.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 and flags that were used during the testing process. 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 39: WrappedERC4626CurvePool.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

H WrappedERC4626YearnV2Vault.sol

H.1 Race Condition In pricePerShare [HIGH]

Description:

The _convertToShares and _convertToAssets functions use the pricePerShare from Yearn contract. The issue here is that there is a big probability to have a desynchronization between what the user is seeing using the view function in Yearn and the real amount calculated after the transaction is executed, causing the user to make a decision based on some wrong inputs.

Code:

Listing 40: WrappedERC4626YearnV2Vault.sol

```
30  function _convertToShares(uint256 assets, Math.Rounding rounding)
        ↪ internal view virtual override returns (uint256 shares)
31  {
32   return assets / vault.pricePerShare();
33  }
```

Listing 41: WrappedERC4626YearnV2Vault.sol

```
36  function _convertToAssets(uint256 shares, Math.Rounding rounding)
        ↪ internal view virtual override returns (uint256 assets)
37  {
38   return shares * vault.pricePerShare();
39  }
```

Risk Level:

Likelihood – 3
Impact - 5

Recommendation:

We recommend adding the value of pricePerShare from the front in the parameter of functions that call _convertToShares and _convertToAssets and verify if the two values match (the value in the front and the vault.pricePerShare() if it's not the case revert the transaction.)

Status - Mitigated

The team has mitigated the risk by using static calls to get real precision.

H.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, some of the contract's functionality may become inaccessible.

Code:

Listing 42: WrappedERC4626YearnV2Vault.sol

```
23  constructor (IERC20Metadata asset_, string memory name, string memory
        ↪ symbol, YeanV2VaultAPI _vault) ERC4626(asset_) ERC20(name, symbol
        ↪ )
24  {
25   vault = _vault;
26   IERC20 token = IERC20(vault.token());
27   token.approve(address(vault), type(uint256).max);
28  }
```

Risk Level:

Likelihood – 1

Impact - 3

Recommendation:

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

Status - Fixed

The team has resolved the issue by adding the necessary verifications.

Code:

Listing 43: WrappedERC4626YearnV2Vault.sol

```
27      constructor(
28          IYearnV2Vault _vault,
29          string memory name,
30          string memory symbol
31      ) ERC4626(IERC20Metadata(_vault.token())) ERC20(name, symbol) {
32          require(address(_vault) != address(0), "Zero address provided");
33          vault = _vault;
34          IERC20(asset()).safeIncreaseAllowance(
35              address(_vault),
36              type(uint256).max
37          );
```

H.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 and flags that were used during the testing process. 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 44: WrappedERC4626YearnV2Vault.sol

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

Risk Level:

Likelihood – 2
Impact - 2

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 has resolved the issue by locking the pragma.

4 Best Practices

BP.1 Dead Code

Description:

In the CurveConverter contract, the swap function is empty and doesn't return anything. Therefore, it represents dead code. Consider removing the function or add logic to it.
Same issue in ApyFlow contract in the rebalance function (L119).

Code:

Listing 45: CurveConverter (Line 14)

```
1 function swap(address source, address destination, uint256 value,
      ↪ address beneficiary) external returns (uint256)
```

Listing 46: ApyFlow (Line 118)

```
1   // todo: is it need?
2   //assetsInSingleAssetVault = assetsInSingleAssetVault - assets + value
        ↪ ;
```

BP.2 Missing Error Message

Description:

In the ApyFlow contract, the addNewVault function verifies if the vault already exists using the require(address(vaultsForToken[token].vault) == address(0)), it's advised if the condition is not met to return an error message to the owner to give him the context of the error.

Code:

Listing 47: CurveConverter (Line 14)

```
1 function swap(address source, address destination, uint256 value,
      ↪ address beneficiary) external returns (uint256)
```

5 Tests

Results:

```
Brownie v1.19.1 - Python development framework for Ethereum

============================= test session starts
    ↪ ==============================
platform linux -- Python 3.8.10, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: /home/shellboxes/Downloads/apyflow_updated
plugins: eth-brownie-1.19.1, hypothesis-6.27.3, xdist-1.34.0, web3
    ↪ -5.30.0, forked-1.4.0
collected 6 items

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

tests/test_deploy.py .... [ 66%]
tests/test_mocks/test_yearn.py .. [100%]

============================== warnings summary
    ↪ ===============================
tests/test_deploy.py: 66 warnings
tests/test_mocks/test_yearn.py: 25 warnings
  /home/shellboxes/.local/pipx/venvs/eth-brownie/lib/python3.8/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/test_deploy.py: 18 warnings
tests/test_mocks/test_yearn.py: 9 warnings
  /home/shellboxes/.local/pipx/venvs/eth-brownie/lib/python3.8/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
======================= 6 passed, 118 warnings in 15.07s
   ↪ =======================
Terminating local RPC client...
```

6 Coverage

Results:

```
Brownie v1.19.1 - Python development framework for Ethereum

============================= test session starts
    ↪ ==============================
platform linux -- Python 3.8.10, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: /home/shellboxes/Downloads/apyflow_updated
plugins: eth-brownie-1.19.1, hypothesis-6.27.3, xdist-1.34.0, web3
    ↪ -5.30.0, forked-1.4.0
collected 6 items

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

tests/test_deploy.py .... [ 66%]
tests/test_mocks/test_yearn.py .. [100%]

============================== warnings summary
    ↪ ===============================
tests/test_deploy.py: 84 warnings
tests/test_mocks/test_yearn.py: 32 warnings
  /home/shellboxes/.local/pipx/venvs/eth-brownie/lib/python3.8/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/test_deploy.py: 282 warnings
tests/test_mocks/test_yearn.py: 32 warnings
  /home/shellboxes/.local/pipx/venvs/eth-brownie/lib/python3.8/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
================================== Coverage
    ↪ ==================================

  contract: ApyFlow - 20.0%
    ApyFlow.addNewVault - 75.0%
    ERC20._mint - 75.0%
    Ownable._checkOwner - 75.0%
    ApyFlow.rebalance - 0.0%
    ERC20._approve - 0.0%
    ERC20._burn - 0.0%
    ERC20._spendAllowance - 0.0%
    ERC20._transfer - 0.0%
    ERC20.decreaseAllowance - 0.0%
    Ownable.transferOwnership - 0.0%

  contract: MockPortfolioScore - 29.2%
    Ownable._checkOwner - 75.0%
    Ownable.transferOwnership - 0.0%

  contract: Token - 52.5%
    ERC20._spendAllowance - 87.5%
    ERC20._approve - 75.0%
    ERC20._transfer - 75.0%
    ERC20._burn - 0.0%
    ERC20.decreaseAllowance - 0.0%

  contract: YearnMock - 30.4%
    YearnMock.pricePerShare - 100.0%
```

```
    ERC20._burn - 75.0%
    ERC20._mint - 75.0%
    ERC20._approve - 0.0%
    ERC20._spendAllowance - 0.0%
    ERC20._transfer - 0.0%
    ERC20.decreaseAllowance - 0.0%

Coverage report saved at /home/shellboxes/Downloads/apyflow_updated/
    ↪ reports/coverage.json
View the report using the Brownie GUI
====================== 6 passed, 430 warnings in 30.54s
    ↪ ======================
Terminating local RPC client...
```

Conclusion:

The code coverage results were obtained by running npx hardhat coverage. We found the line coverage to be 33.02%. We find the code coverage to be dangerously low and recommend raising the line coverage to 100%. This is particularly important since APYFlow protocol contains logic for rebalance and vaults and as well as complex smart contract interactions. We also recommend adding integration tests using a mainnet fork.

7 Coverage Update

Results:

```
contract: ApyFlow - 40.9%
          ApyFlow.convertToShares - 100.0%
          ApyFlow.addNewVault - 75.0%
          ApyFlow.convertToAssets - 75.0%
          ApyFlow.deposit - 75.0%
          ApyFlow.rebalance - 75.0%
          ERC20._mint - 75.0%
          Ownable._checkOwner - 75.0%
```

```
            SafeERC20._callOptionalReturn - 75.0%
            Address.verifyCallResult - 37.5%
            ERC20._burn - 8.3%
            ERC20._approve - 0.0%
            ERC20._spendAllowance - 0.0%
            ERC20._transfer - 0.0%
            ERC20.decreaseAllowance - 0.0%
            Ownable.transferOwnership - 0.0%

contract: ConverterMock - 0.0%

contract: MockPortfolioScore - 29.2%
            Ownable._checkOwner - 75.0%
            Ownable.transferOwnership - 0.0%

contract: Token - 52.5%
            ERC20._spendAllowance - 87.5%
            ERC20._approve - 75.0%
            ERC20._transfer - 75.0%
            ERC20._burn - 0.0%
            ERC20.decreaseAllowance - 0.0%

contract: YearnMock - 36.7%
            YearnMock.pricePerShare - 100.0%
            Address.functionCallWithValue - 75.0%
            ERC20._burn - 75.0%
            ERC20._mint - 75.0%
            SafeERC20._callOptionalReturn - 75.0%
            Address.verifyCallResult - 37.5%
            ERC20._approve - 0.0%
            ERC20._spendAllowance - 0.0%
            ERC20._transfer - 0.0%
            ERC20.decreaseAllowance - 0.0%
```

```
contract: AssetConverter - 58.2%
          AssetConverter.updateConverter - 100.0%
          Address.functionCallWithValue - 75.0%
          Ownable._checkOwner - 75.0%
          SafeERC20._callOptionalReturn - 75.0%
          Address.verifyCallResult - 37.5%
          Ownable.transferOwnership - 0.0%

contract: UniswapV2Converter - 61.7%
          UniswapV2Converter.swap - 100.0%
          Address.functionCallWithValue - 75.0%
          SafeERC20._callOptionalReturn - 75.0%
          Address.verifyCallResult - 37.5%
          Ownable._checkOwner - 0.0%
          Ownable.transferOwnership - 0.0%

contract: UniswapV3Converter - 60.1%
          Address.functionCallWithValue - 75.0%
          SafeERC20._callOptionalReturn - 75.0%
          UniswapV3Converter.swap - 75.0%
          Address.verifyCallResult - 37.5%
```

Conclusion:

After the re-audit, the code coverage results were improved to 48.47%, the integration tests were added using the mainnet fork.

8 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:

```
'npx hardhat compile --force' running
Compiled 48 Solidity files successfully

Warning: Unused function parameter. Remove or comment out the variable
   ↪ name to silence this warning.
   --> contracts/SingleAssetVault.sol:102:44:
    |
102 |  function _convertToShares(uint256 assets, Math.Rounding rounding)
   ↪  internal view virtual override returns (uint256 shares)
    |  ^^^^^^^^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
   ↪ name to silence this warning.
   --> contracts/SingleAssetVault.sol:107:44:
    |
107 |  function _convertToAssets(uint256 shares, Math.Rounding rounding)
   ↪  internal view virtual override returns (uint256 assets)
    |  ^^^^^^^^^^^^^^^^^^^^^^
```

```
Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/mocks/CurveMock.sol:24:15:
   |
24 |  constructor (address _token, string memory name, string memory
    ↪ symbol)
   |  ^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/mocks/CurveMock.sol:34:58:
   |
34 |  function calc_token_amount(uint256[3] calldata amounts, bool
    ↪ is_deposit) external returns (uint256)
   |  ^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/mocks/CurveMock.sol:39:54:
   |
39 |  function add_liquidity(uint256[3] calldata amounts, uint256
    ↪ min_mint_amount) external returns (uint256)
   |  ^^^^^^^^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/mocks/CurveMock.sol:47:59:
   |
47 |  function remove_liquidity_one_coin(uint256 token_amount, uint256
    ↪ index, uint256 min_amount) external returns (uint256)
   |  ^^^^^^^^^^^^^
```

```
Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/mocks/CurveMock.sol:47:74:
   |
47 |  function remove_liquidity_one_coin(uint256 token_amount, uint256
    ↪ index, uint256 min_amount) external returns (uint256)
   |  ^^^^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/protocol-vaults/WrappedERC4626CurvePool.sol:38:44:
   |
38 |  function _convertToShares(uint256 assets, Math.Rounding rounding)
    ↪ internal view virtual override returns (uint256 shares)
   |  ^^^^^^^^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/protocol-vaults/WrappedERC4626CurvePool.sol:43:44:
   |
43 |  function _convertToAssets(uint256 shares, Math.Rounding rounding)
    ↪ internal view virtual override returns (uint256 assets)
   |  ^^^^^^^^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
    ↪ name to silence this warning.
  --> contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol:31:44:
   |
```

```
31 |  function _convertToShares(uint256 assets, Math.Rounding rounding)
   ↪ internal view virtual override returns (uint256 shares)
   |  ^^^^^^^^^^^^^^^^^^^^^^


Warning: Unused function parameter. Remove or comment out the variable
   ↪ name to silence this warning.
  --> contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol:36:44:
   |
36 |  function _convertToAssets(uint256 shares, Math.Rounding rounding)
   ↪ internal view virtual override returns (uint256 assets)
   |  ^^^^^^^^^^^^^^^^^^^^^^


Warning: Function state mutability can be restricted to view
  --> contracts/mocks/CurveMock.sol:34:2:
   |
34 |  function calc_token_amount(uint256[3] calldata amounts, bool
   ↪ is_deposit) external returns (uint256)
   |  ^ (Relevant source part starts here and spans across multiple
      ↪ lines).



ICurvePool is re-used:
 - ICurvePool (contracts/mocks/CurveMock.sol#7-16)
 - ICurvePool (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #name-reused

ApyFlow.deposit(address,uint256) (contracts/ApyFlow.sol#78-86) ignores
   ↪ return value by IERC20(token).transferFrom(msg.sender,address(
```

```
    ↪ this),value) (contracts/ApyFlow.sol#80)
AssetConverter.swap(address,address,uint256) (contracts/AssetConverter.
    ↪ sol#31-35) ignores return value by IERC20(source).transferFrom(
    ↪ msg.sender,address(this),value) (contracts/AssetConverter.sol#33)
SingleAssetVault._deposit(address,address,uint256,uint256) (contracts/
    ↪ SingleAssetVault.sol#117-134) ignores return value by token.
    ↪ transferFrom(caller,address(this),assets) (contracts/
    ↪ SingleAssetVault.sol#125)
CurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/CurveMock.
    ↪ sol#39-45) ignores return value by token.transferFrom(msg.sender,
    ↪ address(this),amounts[0]) (contracts/mocks/CurveMock.sol#41)
CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (contracts/
    ↪ mocks/CurveMock.sol#47-55) ignores return value by lp_token.
    ↪ transferFrom(msg.sender,address(this),token_amount) (contracts/
    ↪ mocks/CurveMock.sol#49)
CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (contracts/
    ↪ mocks/CurveMock.sol#47-55) ignores return value by token.transfer
    ↪ (msg.sender,assets) (contracts/mocks/CurveMock.sol#52)
YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol#23-30) ignores
    ↪  return value by token.transferFrom(msg.sender,address(this),
    ↪ amount) (contracts/mocks/YearnMock.sol#26)
YearnMock.withdraw(uint256,address) (contracts/mocks/YearnMock.sol
    ↪ #32-39) ignores return value by token.transfer(recipient,tokens)
    ↪ (contracts/mocks/YearnMock.sol#35)
WrappedERC4626CurvePool._deposit(address,address,uint256,uint256) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#53-66)
    ↪ ignores return value by token.transferFrom(caller,address(this),
    ↪ assets) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #60)
WrappedERC4626YearnV2Vault._deposit(address,address,uint256,uint256) (
    ↪ contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol#46-59)
    ↪ ignores return value by token.transferFrom(caller,address(this),
    ↪ assets) (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol
    ↪ #54)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unchecked-transfer

PortfolioScore.scores (contracts/PortfolioScore.sol#13) is never
    ↪ initialized. It is used in:
 - PortfolioScore.getPortfolioScore(address) (contracts/PortfolioScore.
     ↪ sol#21-33)
CurvePool.token (contracts/mocks/CurveMock.sol#22) is never initialized.
    ↪  It is used in:
 - CurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/
     ↪ CurveMock.sol#39-45)
 - CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (
     ↪ contracts/mocks/CurveMock.sol#47-55)
 - CurvePool.convertToShares(uint256) (contracts/mocks/CurveMock.sol
     ↪ #57-60)
 - CurvePool.convertToAssets(uint256) (contracts/mocks/CurveMock.sol
     ↪ #62-65)
 - CurvePool.pricePerToken() (contracts/mocks/CurveMock.sol#67-70)
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)
PortfolioScore.getPortfolioScore(address) (contracts/PortfolioScore.sol
   ↪ #21-33) performs a multiplication on the result of a division:
 -riskScore /= 100 (contracts/PortfolioScore.sol#30)
 -(riskScore * 60 + scoreData.profitScore * 40) / 100 (contracts/
    ↪ PortfolioScore.sol#32)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #divide-before-multiply

ERC4626._isVaultCollateralized() (node_modules/@openzeppelin/contracts/
   ↪ token/ERC20/extensions/ERC4626.sol#219-221) uses a dangerous
   ↪ strict equality:
 - totalAssets() > 0 || totalSupply() == 0 (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/extensions/ERC4626.sol#220)
SingleAssetVault.pricePerShare() (contracts/SingleAssetVault.sol#45-57)
   ↪ uses a dangerous strict equality:
 - totalSupply() == 0 (contracts/SingleAssetVault.sol#56)
YearnMock.pricePerShare() (contracts/mocks/YearnMock.sol#18-21) uses a
   ↪ dangerous strict equality:
 - totalSupply() == 0 (contracts/mocks/YearnMock.sol#20)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #dangerous-strict-equalities
```

```
Reentrancy in SingleAssetVault._deposit(address,address,uint256,uint256)
    ↪  (contracts/SingleAssetVault.sol#117-134):
 External calls:
 - token.transferFrom(caller,address(this),assets) (contracts/
    ↪ SingleAssetVault.sol#125)
 - vaultInfo.vault.deposit(assets,address(this)) (contracts/
    ↪ SingleAssetVault.sol#128)
 State variables written after the call(s):
 - assetsInSingleAssetVault += assets (contracts/SingleAssetVault.sol
    ↪ #129)
 - vaultInfo.assetsInVault += assets (contracts/SingleAssetVault.sol
    ↪ #130)
Reentrancy in SingleAssetVault._withdraw(address,address,address,uint256
    ↪ ,uint256) (contracts/SingleAssetVault.sol#136-152):
 External calls:
 - vaultInfo.vault.withdraw(assets,msg.sender,address(this)) (contracts/
    ↪ SingleAssetVault.sol#147)
 State variables written after the call(s):
 - assetsInSingleAssetVault -= assets (contracts/SingleAssetVault.sol
    ↪ #148)
 - vaultInfo.assetsInVault -= assets (contracts/SingleAssetVault.sol
    ↪ #149)
Reentrancy in YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol
    ↪ #23-30):
 External calls:
 - token.transferFrom(msg.sender,address(this),amount) (contracts/mocks/
    ↪ YearnMock.sol#26)
 State variables written after the call(s):
 - _mint(msg.sender,shares) (contracts/mocks/YearnMock.sol#27)
  - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/
     ↪ ERC20/ERC20.sol#262)
Reentrancy in SingleAssetVault.rebalance(uint256,uint256,uint256) (
    ↪ contracts/SingleAssetVault.sol#85-100):
 External calls:
```

```
 - value = vaults[fromVault].vault.withdraw(assets,address(this),address
     ↪ (this)) (contracts/SingleAssetVault.sol#92)
 - vaults[toVault].vault.deposit(value,address(this)) (contracts/
     ↪ SingleAssetVault.sol#93)
 State variables written after the call(s):
 - assetsInSingleAssetVault = assetsInSingleAssetVault - assets + value
     ↪ (contracts/SingleAssetVault.sol#97)
 - vaults[fromVault].assetsInVault -= assets (contracts/SingleAssetVault
     ↪ .sol#95)
 - vaults[toVault].assetsInVault += value (contracts/SingleAssetVault.
     ↪ sol#96)
Reentrancy in YearnMock.withdraw(uint256,address) (contracts/mocks/
    ↪ YearnMock.sol#32-39):
 External calls:
 - token.transfer(recipient,tokens) (contracts/mocks/YearnMock.sol#35)
 State variables written after the call(s):
 - _burn(msg.sender,amount) (contracts/mocks/YearnMock.sol#36)
  - _totalSupply -= amount (node_modules/@openzeppelin/contracts/token/
      ↪ ERC20/ERC20.sol#290)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-1

ChainlinkClient.buildOperatorRequest(bytes32,bytes4).req (node_modules/
    ↪ @chainlink/contracts/src/v0.8/ChainlinkClient.sol#67) 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.buildChainlinkRequest(bytes32,address,bytes4).req (
    ↪ node_modules/@chainlink/contracts/src/v0.8/ChainlinkClient.sol
    ↪ #52) 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.addNewVault(address,address,uint256) (contracts/ApyFlow.sol
    ↪ #46-56) ignores return value by IERC20(token).approve(vault,type
    ↪ ()(uint256).max) (contracts/ApyFlow.sol#51)
ApyFlow.addNewVault(address,address,uint256) (contracts/ApyFlow.sol
    ↪ #46-56) ignores return value by IERC20(token).approve(address(
    ↪ assetConverter),type()(uint256).max) (contracts/ApyFlow.sol#52)
ApyFlow.deposit(address,uint256) (contracts/ApyFlow.sol#78-86) ignores
    ↪ return value by vaultsForToken[token].vault.deposit(value,address
    ↪ (this)) (contracts/ApyFlow.sol#82)
```

```
ApyFlow.withdraw(address,uint256,address) (contracts/ApyFlow.sol#88-96)
    ↪ ignores return value by vault.withdraw(value,recipient,address(
    ↪ this)) (contracts/ApyFlow.sol#93)
ApyFlow.rebalance(address,address,uint256) (contracts/ApyFlow.sol
    ↪ #107-123) ignores return value by vaultsForToken[destinationToken
    ↪ ].vault.deposit(newValue,address(this)) (contracts/ApyFlow.sol
    ↪ #117)
AssetConverter.updateConverter(address,address,address) (contracts/
    ↪ AssetConverter.sol#25-29) ignores return value by IERC20(source).
    ↪ approve(newConverter,type()(uint256).max) (contracts/
    ↪ AssetConverter.sol#27)
SingleAssetVault.addNewVault(IERC4626) (contracts/SingleAssetVault.sol
    ↪ #37-43) ignores return value by IERC20(asset()).approve(address(
    ↪ vault),type()(uint256).max) (contracts/SingleAssetVault.sol#41)
SingleAssetVault.rebalance(uint256,uint256,uint256) (contracts/
    ↪ SingleAssetVault.sol#85-100) ignores return value by vaults[
    ↪ toVault].vault.deposit(value,address(this)) (contracts/
    ↪ SingleAssetVault.sol#93)
SingleAssetVault._deposit(address,address,uint256,uint256) (contracts/
    ↪ SingleAssetVault.sol#117-134) ignores return value by vaultInfo.
    ↪ vault.deposit(assets,address(this)) (contracts/SingleAssetVault.
    ↪ sol#128)
SingleAssetVault._withdraw(address,address,address,uint256,uint256) (
    ↪ contracts/SingleAssetVault.sol#136-152) ignores return value by
    ↪ vaultInfo.vault.withdraw(assets,msg.sender,address(this)) (
    ↪ contracts/SingleAssetVault.sol#147)
WrappedERC4626CurvePool.constructor(address,address,address,string,
    ↪ string) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #28-36) ignores return value by token.approve(curve,type()(
    ↪ uint256).max) (contracts/protocol-vaults/WrappedERC4626CurvePool.
    ↪ sol#34)
WrappedERC4626CurvePool.constructor(address,address,address,string,
    ↪ string) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #28-36) ignores return value by lpToken.approve(curve,type()(
```

```
    ↪ uint256).max) (contracts/protocol-vaults/WrappedERC4626CurvePool.
    ↪ sol#35)
WrappedERC4626CurvePool._deposit(address,address,uint256,uint256) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#53-66)
    ↪ ignores return value by curvePool.add_liquidity(amounts,shares) (
    ↪ contracts/protocol-vaults/WrappedERC4626CurvePool.sol#62)
WrappedERC4626CurvePool._withdraw(address,address,address,uint256,
    ↪ uint256) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #68-80) ignores return value by curvePool.
    ↪ remove_liquidity_one_coin(shares,0,shares) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePool.sol#77)
WrappedERC4626YearnV2Vault.constructor(IERC20Metadata,string,string,
    ↪ YeanV2VaultAPI) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#24-29) ignores return value by
    ↪ token.approve(address(vault),type()(uint256).max) (contracts/
    ↪ protocol-vaults/WrappedERC4626YearnV2Vault.sol#28)
WrappedERC4626YearnV2Vault._deposit(address,address,uint256,uint256) (
    ↪ contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol#46-59)
    ↪ ignores return value by vault.deposit(assets) (contracts/protocol
    ↪ -vaults/WrappedERC4626YearnV2Vault.sol#55)
WrappedERC4626YearnV2Vault._withdraw(address,address,address,uint256,
    ↪ uint256) (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.
    ↪ sol#61-73) ignores return value by vault.withdraw(shares,msg.
    ↪ sender) (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol
    ↪ #70)
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).name
    ↪ (contracts/SingleAssetVault.sol#31) 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).
    ↪ symbol (contracts/SingleAssetVault.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)
SingleAssetVault._withdraw(address,address,address,uint256,uint256).
    ↪ owner (contracts/SingleAssetVault.sol#139) shadows:
 - Ownable.owner() (node_modules/@openzeppelin/contracts/access/Ownable.
     ↪ sol#43-45) (function)
Token.constructor(string,string,uint256).name (contracts/mocks/Token.sol
    ↪ #10) 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).symbol (contracts/mocks/Token.
    ↪ sol#10) 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)._name (contracts/mocks/
   ↪ YearnMock.sol#13) shadows:
 - ERC20._name (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.
    ↪ sol#42) (state variable)
YearnMock.constructor(address,string,string)._symbol (contracts/mocks/
   ↪ YearnMock.sol#13) shadows:
 - ERC20._symbol (node_modules/@openzeppelin/contracts/token/ERC20/ERC20
    ↪ .sol#43) (state variable)
WrappedERC4626CurvePool.constructor(address,address,address,string,
   ↪ string).asset (contracts/protocol-vaults/WrappedERC4626CurvePool.
   ↪ sol#28) shadows:
 - ERC4626.asset() (node_modules/@openzeppelin/contracts/token/ERC20/
    ↪ extensions/ERC4626.sol#39-41) (function)
 - IERC4626.asset() (node_modules/@openzeppelin/contracts/interfaces/
    ↪ IERC4626.sol#32) (function)
WrappedERC4626CurvePool.constructor(address,address,address,string,
   ↪ string).name (contracts/protocol-vaults/WrappedERC4626CurvePool.
   ↪ 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)
WrappedERC4626CurvePool.constructor(address,address,address,string,
   ↪ string).symbol (contracts/protocol-vaults/WrappedERC4626CurvePool
   ↪ .sol#28) 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(IERC20Metadata,string,string,
   ↪ YeanV2VaultAPI).name (contracts/protocol-vaults/
   ↪ WrappedERC4626YearnV2Vault.sol#24) 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(IERC20Metadata,string,string,
   ↪ YeanV2VaultAPI).symbol (contracts/protocol-vaults/
   ↪ WrappedERC4626YearnV2Vault.sol#24) 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

SingleAssetVault.addNewVault(IERC4626) (contracts/SingleAssetVault.sol
   ↪ #37-43) should emit an event for:
 - totalPortfolioScore += portfolioScore (contracts/SingleAssetVault.sol
    ↪ #42)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #missing-events-arithmetic

ApyFlow.recomputePricePerToken() (contracts/ApyFlow.sol#58-76) has
   ↪ external calls inside a loop: shares = vaultInfo.vault.balanceOf(
   ↪ address(this)) (contracts/ApyFlow.sol#67)
ApyFlow.recomputePricePerToken() (contracts/ApyFlow.sol#58-76) has
   ↪ external calls inside a loop: balanceAtVault = vaultInfo.vault.
   ↪ convertToAssets(shares) (contracts/ApyFlow.sol#68)
ApyFlow.recomputePricePerToken() (contracts/ApyFlow.sol#58-76) has
   ↪ external calls inside a loop: portfolioScore += vaultInfo.vault.
   ↪ totalPortfolioScore() (contracts/ApyFlow.sol#70)
SingleAssetVault.pricePerShare() (contracts/SingleAssetVault.sol#45-57)
   ↪ has external calls inside a loop: shares = vaults[j].vault.
   ↪ balanceOf(address(this)) (contracts/SingleAssetVault.sol#51)
```

```
SingleAssetVault.pricePerShare() (contracts/SingleAssetVault.sol#45-57)
    ↪ has external calls inside a loop: balanceAtVault = vaults[j].
    ↪ vault.convertToAssets(shares) (contracts/SingleAssetVault.sol#52)
SingleAssetVault.computeScoreDeviationInPpm(uint256) (contracts/
    ↪ SingleAssetVault.sol#79-83) has external calls inside a loop:
    ↪ portfolioScore = oracle.getPortfolioScore(address(vaults[
    ↪ vaultIndex].vault)) (contracts/SingleAssetVault.sol#81)
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#117-134):
 External calls:
 - token.transferFrom(caller,address(this),assets) (contracts/
     ↪ SingleAssetVault.sol#125)
 - vaultInfo.vault.deposit(assets,address(this)) (contracts/
     ↪ SingleAssetVault.sol#128)
 State variables written after the call(s):
```

```
 - _mint(receiver,shares) (contracts/SingleAssetVault.sol#131)
  - _balances[account] += amount (node_modules/@openzeppelin/contracts/
    ↪ token/ERC20/ERC20.sol#263)
 - _mint(receiver,shares) (contracts/SingleAssetVault.sol#131)
  - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/
    ↪ ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626CurvePool._deposit(address,address,uint256,
   ↪ uint256) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
   ↪ #53-66):
 External calls:
 - token.transferFrom(caller,address(this),assets) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePool.sol#60)
 - curvePool.add_liquidity(amounts,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#62)
 State variables written after the call(s):
 - _mint(receiver,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#63)
  - _balances[account] += amount (node_modules/@openzeppelin/contracts/
    ↪ token/ERC20/ERC20.sol#263)
 - _mint(receiver,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#63)
  - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/
    ↪ ERC20/ERC20.sol#262)
Reentrancy in WrappedERC4626YearnV2Vault._deposit(address,address,
   ↪ uint256,uint256) (contracts/protocol-vaults/
   ↪ WrappedERC4626YearnV2Vault.sol#46-59):
 External calls:
 - token.transferFrom(caller,address(this),assets) (contracts/protocol-
    ↪ vaults/WrappedERC4626YearnV2Vault.sol#54)
 - vault.deposit(assets) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#55)
 State variables written after the call(s):
 - _mint(receiver,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#56)
```

```
  - _balances[account] += amount (node_modules/@openzeppelin/contracts/
      ↪ token/ERC20/ERC20.sol#263)
 - _mint(receiver,shares) (contracts/protocol-vaults/
     ↪ WrappedERC4626YearnV2Vault.sol#56)
  - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/
      ↪ ERC20/ERC20.sol#262)
Reentrancy in ApyFlow.addNewVault(address,address,uint256) (contracts/
    ↪ ApyFlow.sol#46-56):
 External calls:
 - IERC20(token).approve(vault,type()(uint256).max) (contracts/ApyFlow.
     ↪ sol#51)
 - IERC20(token).approve(address(assetConverter),type()(uint256).max) (
     ↪ contracts/ApyFlow.sol#52)
 State variables written after the call(s):
 - tokens.push(token) (contracts/ApyFlow.sol#53)
Reentrancy in SingleAssetVault.addNewVault(IERC4626) (contracts/
    ↪ SingleAssetVault.sol#37-43):
 External calls:
 - IERC20(asset()).approve(address(vault),type()(uint256).max) (
     ↪ contracts/SingleAssetVault.sol#41)
 State variables written after the call(s):
 - totalPortfolioScore += portfolioScore (contracts/SingleAssetVault.sol
     ↪ #42)
Reentrancy in ApyFlow.deposit(address,uint256) (contracts/ApyFlow.sol
    ↪ #78-86):
 External calls:
 - IERC20(token).transferFrom(msg.sender,address(this),value) (contracts
     ↪ /ApyFlow.sol#80)
 - vaultsForToken[token].vault.deposit(value,address(this)) (contracts/
     ↪ ApyFlow.sol#82)
 State variables written after the call(s):
 - _mint(msg.sender,tokensNumber) (contracts/ApyFlow.sol#83)
  - _balances[account] += amount (node_modules/@openzeppelin/contracts/
      ↪ token/ERC20/ERC20.sol#263)
```

```
 - _mint(msg.sender,tokensNumber) (contracts/ApyFlow.sol#83)
  - _totalSupply += amount (node_modules/@openzeppelin/contracts/token/
    ↪ ERC20/ERC20.sol#262)
Reentrancy in YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol
   ↪ #23-30):
 External calls:
 - token.transferFrom(msg.sender,address(this),amount) (contracts/mocks/
    ↪ YearnMock.sol#26)
 State variables written after the call(s):
 - _mint(msg.sender,shares) (contracts/mocks/YearnMock.sol#27)
  - _balances[account] += amount (node_modules/@openzeppelin/contracts/
    ↪ token/ERC20/ERC20.sol#263)
Reentrancy in AssetConverter.updateConverter(address,address,address) (
   ↪ contracts/AssetConverter.sol#25-29):
 External calls:
 - IERC20(source).approve(newConverter,type()(uint256).max) (contracts/
    ↪ AssetConverter.sol#27)
 State variables written after the call(s):
 - converters[source][destination] = IConverter(newConverter) (contracts
    ↪ /AssetConverter.sol#28)
Reentrancy in YearnMock.withdraw(uint256,address) (contracts/mocks/
   ↪ YearnMock.sol#32-39):
 External calls:
 - token.transfer(recipient,tokens) (contracts/mocks/YearnMock.sol#35)
 State variables written after the call(s):
 - _burn(msg.sender,amount) (contracts/mocks/YearnMock.sol#36)
  - _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#117-134):
External calls:
- token.transferFrom(caller,address(this),assets) (contracts/
    ↪ SingleAssetVault.sol#125)
- vaultInfo.vault.deposit(assets,address(this)) (contracts/
    ↪ SingleAssetVault.sol#128)
Event emitted after the call(s):
- Deposit(caller,receiver,assets,shares) (contracts/SingleAssetVault.
    ↪ sol#133)
- Transfer(address(0),account,amount) (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/ERC20.sol#264)
  - _mint(receiver,shares) (contracts/SingleAssetVault.sol#131)
Reentrancy in WrappedERC4626CurvePool._deposit(address,address,uint256,
    ↪ uint256) (contracts/protocol-vaults/WrappedERC4626CurvePool.sol
    ↪ #53-66):
External calls:
- token.transferFrom(caller,address(this),assets) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePool.sol#60)
- curvePool.add_liquidity(amounts,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#62)
Event emitted after the call(s):
```

```
 - Deposit(caller,receiver,assets,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626CurvePool.sol#65)
 - Transfer(address(0),account,amount) (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/ERC20.sol#264)
  - _mint(receiver,shares) (contracts/protocol-vaults/
     ↪ WrappedERC4626CurvePool.sol#63)
Reentrancy in WrappedERC4626YearnV2Vault._deposit(address,address,
   ↪ uint256,uint256) (contracts/protocol-vaults/
   ↪ WrappedERC4626YearnV2Vault.sol#46-59):
 External calls:
 - token.transferFrom(caller,address(this),assets) (contracts/protocol-
    ↪ vaults/WrappedERC4626YearnV2Vault.sol#54)
 - vault.deposit(assets) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#55)
 Event emitted after the call(s):
 - Deposit(caller,receiver,assets,shares) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#58)
 - Transfer(address(0),account,amount) (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/ERC20.sol#264)
  - _mint(receiver,shares) (contracts/protocol-vaults/
     ↪ WrappedERC4626YearnV2Vault.sol#56)
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#136-152):
 External calls:
```

```
 - vaultInfo.vault.withdraw(assets,msg.sender,address(this)) (contracts/
    ↪ SingleAssetVault.sol#147)
 Event emitted after the call(s):
 - Withdraw(caller,receiver,owner,assets,shares) (contracts/
    ↪ SingleAssetVault.sol#151)
Reentrancy in WrappedERC4626CurvePool._withdraw(address,address,address,
   ↪ uint256,uint256) (contracts/protocol-vaults/
   ↪ WrappedERC4626CurvePool.sol#68-80):
 External calls:
 - curvePool.remove_liquidity_one_coin(shares,0,shares) (contracts/
    ↪ protocol-vaults/WrappedERC4626CurvePool.sol#77)
 Event emitted after the call(s):
 - Withdraw(caller,receiver,owner,assets,shares) (contracts/protocol-
    ↪ vaults/WrappedERC4626CurvePool.sol#79)
Reentrancy in WrappedERC4626YearnV2Vault._withdraw(address,address,
   ↪ address,uint256,uint256) (contracts/protocol-vaults/
   ↪ WrappedERC4626YearnV2Vault.sol#61-73):
 External calls:
 - vault.withdraw(shares,msg.sender) (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#70)
 Event emitted after the call(s):
 - Withdraw(caller,receiver,owner,assets,shares) (contracts/protocol-
    ↪ vaults/WrappedERC4626YearnV2Vault.sol#72)
Reentrancy in ApyFlow.addNewVault(address,address,uint256) (contracts/
   ↪ ApyFlow.sol#46-56):
 External calls:
 - IERC20(token).approve(vault,type()(uint256).max) (contracts/ApyFlow.
    ↪ sol#51)
 - IERC20(token).approve(address(assetConverter),type()(uint256).max) (
    ↪ contracts/ApyFlow.sol#52)
 Event emitted after the call(s):
 - NewVaultAdded(token) (contracts/ApyFlow.sol#55)
Reentrancy in ApyFlow.deposit(address,uint256) (contracts/ApyFlow.sol
   ↪ #78-86):
```

```
 External calls:
 - IERC20(token).transferFrom(msg.sender,address(this),value) (contracts
    ↪ /ApyFlow.sol#80)
 - vaultsForToken[token].vault.deposit(value,address(this)) (contracts/
    ↪ ApyFlow.sol#82)
 Event emitted after the call(s):
 - Deposited(msg.sender,token,value,tokensNumber) (contracts/ApyFlow.sol
    ↪ #85)
 - Transfer(address(0),account,amount) (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/ERC20.sol#264)
  - _mint(msg.sender,tokensNumber) (contracts/ApyFlow.sol#83)
Reentrancy in YearnMock.deposit(uint256) (contracts/mocks/YearnMock.sol
   ↪ #23-30):
 External calls:
 - token.transferFrom(msg.sender,address(this),amount) (contracts/mocks/
    ↪ YearnMock.sol#26)
 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#27)
Reentrancy in ApyFlow.withdraw(address,uint256,address) (contracts/
   ↪ ApyFlow.sol#88-96):
 External calls:
 - vault.withdraw(value,recipient,address(this)) (contracts/ApyFlow.sol
    ↪ #93)
 Event emitted after the call(s):
 - Withdrawal(msg.sender,token,value,tokensNumber) (contracts/ApyFlow.
    ↪ sol#95)
Reentrancy in YearnMock.withdraw(uint256,address) (contracts/mocks/
   ↪ YearnMock.sol#32-39):
 External calls:
 - token.transfer(recipient,tokens) (contracts/mocks/YearnMock.sol#35)
 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#36)
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 is used:
 - Version used: ['>=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)
- >=0.8.0 (contracts/ApyFlow.sol#2)
- >=0.8.0 (contracts/AssetConverter.sol#2)
- >=0.8.0 (contracts/PortfolioScore.sol#2)
- >=0.8.0 (contracts/PortfolioScoreOracle.sol#2)
- >=0.8.0 (contracts/SingleAssetVault.sol#2)
- >=0.8.0 (contracts/converters/CurveConverter.sol#2)
```

```
 - >=0.8.0 (contracts/mocks/CBridgeMock.sol#2)
 - >=0.8.0 (contracts/mocks/CurveMock.sol#2)
 - >=0.8.0 (contracts/mocks/MockPortfolioScore.sol#2)
 - >=0.8.0 (contracts/mocks/Token.sol#2)
 - >=0.8.0 (contracts/mocks/YearnMock.sol#2)
 - >=0.8.0 (contracts/protocol-vaults/WrappedERC4626CurvePool.sol#2)
 - >=0.8.0 (contracts/protocol-vaults/WrappedERC4626YearnV2Vault.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #different-pragma-directives-are-used

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 version>=0.8.0 (contracts/ApyFlow.sol#2) allows old versions
Pragma version>=0.8.0 (contracts/AssetConverter.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/PortfolioScore.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/PortfolioScoreOracle.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/SingleAssetVault.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/converters/CurveConverter.sol#2) allows
   ↪  old versions
Pragma version>=0.8.0 (contracts/mocks/CBridgeMock.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/mocks/CurveMock.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/mocks/MockPortfolioScore.sol#2) allows
   ↪ old versions
Pragma version>=0.8.0 (contracts/mocks/Token.sol#2) allows old versions
Pragma version>=0.8.0 (contracts/mocks/YearnMock.sol#2) allows old
   ↪ versions
Pragma version>=0.8.0 (contracts/protocol-vaults/WrappedERC4626CurvePool
   ↪ .sol#2) allows old versions
```

```
Pragma version>=0.8.0 (contracts/protocol-vaults/
    ↪ WrappedERC4626YearnV2Vault.sol#2) allows old versions
solc-0.8.9 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

Converter (contracts/converters/CurveConverter.sol#14-29) should inherit
    ↪  from IConverter (contracts/AssetConverter.sol#9-12)
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(uint128,uint128,uint256,uint256).min_dy (
    ↪ contracts/converters/CurveConverter.sol#10) is not in mixedCase
Function ICurvePool.get_virtual_price() (contracts/mocks/CurveMock.sol
    ↪ #9) is not in mixedCase
Function ICurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/
    ↪ CurveMock.sol#11) is not in mixedCase
Parameter ICurvePool.add_liquidity(uint256[3],uint256).min_mint_amount (
    ↪ contracts/mocks/CurveMock.sol#11) is not in mixedCase
Function ICurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (
    ↪ contracts/mocks/CurveMock.sol#13) is not in mixedCase
Parameter ICurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ token_amount (contracts/mocks/CurveMock.sol#13) is not in
    ↪ mixedCase
Parameter ICurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ min_amount (contracts/mocks/CurveMock.sol#13) is not in mixedCase
Function ICurvePool.calc_token_amount(uint256[3],bool) (contracts/mocks/
    ↪ CurveMock.sol#15) is not in mixedCase
```

```
Parameter ICurvePool.calc_token_amount(uint256[3],bool).is_deposit (
    ↪ contracts/mocks/CurveMock.sol#15) is not in mixedCase
Function CurvePool.get_virtual_price() (contracts/mocks/CurveMock.sol
    ↪ #29-32) is not in mixedCase
Function CurvePool.calc_token_amount(uint256[3],bool) (contracts/mocks/
    ↪ CurveMock.sol#34-37) is not in mixedCase
Parameter CurvePool.calc_token_amount(uint256[3],bool).is_deposit (
    ↪ contracts/mocks/CurveMock.sol#34) is not in mixedCase
Function CurvePool.add_liquidity(uint256[3],uint256) (contracts/mocks/
    ↪ CurveMock.sol#39-45) is not in mixedCase
Parameter CurvePool.add_liquidity(uint256[3],uint256).min_mint_amount (
    ↪ contracts/mocks/CurveMock.sol#39) is not in mixedCase
Function CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256) (
    ↪ contracts/mocks/CurveMock.sol#47-55) is not in mixedCase
Parameter CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ token_amount (contracts/mocks/CurveMock.sol#47) is not in
    ↪ mixedCase
Parameter CurvePool.remove_liquidity_one_coin(uint256,uint256,uint256).
    ↪ min_amount (contracts/mocks/CurveMock.sol#47) is not in mixedCase
Variable CurvePool.lp_token (contracts/mocks/CurveMock.sol#20) 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#109) is too similar to ApyFlow.rebalance(
    ↪ address,address,uint256).scoreDeviation2 (contracts/ApyFlow.sol
    ↪ #110)
Variable SingleAssetVault.getVaultWithMaxScoreDeviation(bool).
    ↪ scoreDeviation1 (contracts/SingleAssetVault.sol#66) is too
    ↪ similar to SingleAssetVault.getVaultWithMaxScoreDeviation(bool).
    ↪ scoreDeviation2 (contracts/SingleAssetVault.sol#67)
Variable SingleAssetVault.getVaultWithMaxScoreDeviation(bool).
    ↪ scoreDeviation1 (contracts/SingleAssetVault.sol#66) is too
```

```
    ↪ similar to SingleAssetVault.rebalance(uint256,uint256,uint256).
    ↪ scoreDeviation2 (contracts/SingleAssetVault.sol#88)
Variable SingleAssetVault.rebalance(uint256,uint256,uint256).
    ↪ scoreDeviation1 (contracts/SingleAssetVault.sol#87) is too
    ↪ similar to SingleAssetVault.rebalance(uint256,uint256,uint256).
    ↪ scoreDeviation2 (contracts/SingleAssetVault.sol#88)
Variable SingleAssetVault.rebalance(uint256,uint256,uint256).
    ↪ scoreDeviation1 (contracts/SingleAssetVault.sol#87) is too
    ↪ similar to SingleAssetVault.getVaultWithMaxScoreDeviation(bool).
    ↪ scoreDeviation2 (contracts/SingleAssetVault.sol#67)
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 ChainlinkClient (
    ↪ node_modules/@chainlink/contracts/src/v0.8/ChainlinkClient.sol
    ↪ #17-315)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unused-state-variable

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)
convertToAssets(uint256) should be declared external:
 - ERC4626.convertToAssets(uint256) (node_modules/@openzeppelin/
    ↪ contracts/token/ERC20/extensions/ERC4626.sol#54-56)
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)
recomputePricePerToken() should be declared external:
 - ApyFlow.recomputePricePerToken() (contracts/ApyFlow.sol#58-76)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
   ↪ #public-function-that-could-be-declared-external
. analyzed (53 contracts with 77 detectors), 245 result(s) found
```

Conclusion:

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

9 Conclusion

In this audit, we examined the design and implementation of ApyFlow contract and discovered several issues of varying severity. ApyFlow team addressed 18 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.

10 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