QuillAudits

Audit Report
June, 2021

YearnAgnostic

Contents

Introduction

During the period of April 15th, 2021 to June 19th, 2021 – QuillAudits Team performed security audit for YearnAgnostic smart contracts. The code for audit was taken from the following link:

- https://github.com/Yearn-Agnostic/contracts/blob/main/YFIAGToken/TokenYFIAG.sol [Commit ID: 74791b1f2d7ed382bf865069433ddf284537a13a]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/controllers/Controller.sol [Commit ID: f87036e351d3ced1c970509b3e809acb44f561b7]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/dao/Governance.sol [Commit ID: 74791b1f2d7ed382bf865069433ddf284537a13a]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/dao/IRewardDistributionRecipient.sol [Commit ID: 062df66e36e05176ad743f5f809288dcd18c6b78]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/dao/TokenToVotePowerStaking.sol [Commit ID: 74791b1f2d7ed382bf865069433ddf284537a13a]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/dao/VotingPowerFees.sol [Commit ID: f87036e351d3ced1c970509b3e809acb44f561b7]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/dao/VotingPowerFeesAndRewards.sol [Commit ID: f87036e351d3ced1c970509b3e809acb44f561b7]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/interfaces/weth/IWETH.sol [Commit ID: 74791b1f2d7ed382bf865069433ddf284537a13a]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/interfaces/weth/WETH.sol [Commit ID: eac6a75d4a39387730d3112af499f8aba4b13f41]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/vaults/yVault.sol [Commit ID: f87036e351d3ced1c970509b3e809acb44f561b7]

- https://github.com/Yearn-Agnostic/contracts/blob/main/contracts/vaults/yWETH.sol [Commit ID: 74791b1f2d7ed382bf865069433ddf284537a13a]

Updated Contracts

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/YFIAGToken/TokenYFIAG.sol
  [Commit ID: 75cd8dec70a475907b7138a469daab88d50f5a4a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/controllers/Controller.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/dao/Governance.sol
  [Commit ID: 75cd8dec70a475907b7138a469daab88d50f5a4a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/dao/IRewardDistributionRecipient.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/dao/TokenToVotePowerStaking.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/dao/VotingPowerFees.sol
  [Commit ID: 75cd8dec70a475907b7138a469daab88d50f5a4a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/dao/VotingPowerFeesAndRewards.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/interfaces/weth/IWETH.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/interfaces/weth/WETH.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/vaults/yVault.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

- https://github.com/Yearn-Agnostic/contracts/blob/develop/contracts/vaults/yWETH.sol
  [Commit ID: 3ee222570a19f74797fd238eb3ca7b5bc7f4ac1a]

Overview of Contract

YearnAgnostic Finance supports decentralized finance (DeFi) projects deployed on Ethereum blockchain, Binance smart chain, Tron chain etc., focusing on simplicity, user experience, security and privacy.

YearnAgnostic Finance is a DeFi yield aggregator platform providing a classical way to optimally earn yield or maximize profits on assets. YearnAgnostic Finance is a token-based ecosystem, and the underline token is YFIAG Token.

Benefits

- Best way to earn an optimal interest rate on assets.
- Simplifying earning process.
- Community voting for fairness and equity.
- Simple to use by everyone.
- Privacy and anonymous transactions.

Features

- LENDING
- STAKING
- BORROWING
- GOVERNANCE

Tokenomics

- 20% - Staking Rewards
- 40% - Presale
- 5% - Giveway
- 15% - Team & Advisor
- 13% - Marketing & Development
- 7% - Future Development

Details: https://yearnagnostic.finance/

Scope of Audit

The scope of this audit was to analyse YearnAgnostic smart contract codebase for quality, security, and correctness. Following is the list of smart contracts:

- TokenYFIAG.sol
- Controller.sol
- Governance.sol
- IRewardDistributionRecipient.sol
- TokenToVotePowerStaking.sol
- VotingPowerFees.sol

- VotingPowerFeesAndRewards.sol
- IWETH.sol
- WETH.sol
- yVault.sol
- yWETH.sol

OUT-OF-SCOPE: External contracts, External Oracles, other smart contracts in the repository or imported smart contracts, economic attacks.

Checked Vulnerabilities

We have scanned Yearn Agnostic smart contracts for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with (Unexpected) Throw
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level
- Address hardcoded
- Using delete for arrays
- Integer overflow/underflow
- Locked money
- Private modifier
- Revert/require functions
- Using var
- Visibility
- Using blockhash
- Using SHA3
- Using suicide
- Using throw
- Using inline assembly

Techniques and Methods

Throughout the audit of YearnAgnostic smart contracts care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

A combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy regarding the scope of the smart contract audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of smart contracts and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

Structural Analysis
In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure the Smart contract is structured in a way that will not result in future problems.

Code Review / Manual Analysis
Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Static Analysis

Static Analysis of smart contracts was done to identify contract vulnerabilities. In this step series of automated tools are used to test the security of smart contracts.

Gas Consumption

In this step, we have checked the behaviour of smart contract in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Manticore, Slither.

Issue Categories

High severity issues

Issues that must be fixed before deployment else they can create major issues.

Medium level severity issues

These issues will not create major issues in working but affect the performance of the smart contract.

Low level severity issues

These issues are more suggestions that should be implemented to refine the code in terms of gas, fees, speed and code accuracy

Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	4	3
Closed	0	1	3	0

Issues Found – Code Review / Manual Testing

High severity issues

No Issue found under this category

Medium severity issues

1. **Reentrancy**

   One of the major dangers of calling external contracts is that they can take over the control flow, and make changes to your data that the calling function wasn't expecting. It's recommended that the calls to external functions/events should happen after any changes to state variables in your contract, so your contract is not vulnerable to a reentrancy exploit. When control is transferred to the recipient, care must be taken to not create reentrancy vulnerabilities. OpenZeppelin has its own mutex implementation you can use called ReentrancyGuard. This library provides a modifier you can apply to any function called nonReentrant that guards the function with a mutex.
   Consider using ReentrancyGuard or the checks-effects-interactions pattern.

   The following link explains more about Reentrancy and ReentrancyGuard
   https://docs.openzeppelin.com/contracts/2.x/api/utils#ReentrancyGuard
   https://blog.openzeppelin.com/reentrancy-after-istanbul/

   Most Recent DeFi Reentrancy Attack: DFORCE

   Code Lines:
   - yVault.sol: deposit(uint256) [#142-158]
   - yWETH.sol: depositETH() [#21-35]
   - VotingPowerFees.sol: _withdrawFeesFor(address) [#82-98]
   - VotingPowerFeesAndRewards.sol: getReward() [#135-142]
   - Controller.sol: setStrategy(address,address) [#159-167]
   - Governance.sol: withdraw(uint256) [#290-295]

   Auditors Remarks: Fixed

Low level severity issues

1. **The compiler version should be fixed**

   Solidity source files indicate the versions of the compiler they can be compiled with. It's recommended to lock the compiler version in code, as future compiler versions may handle certain language constructions in a way the developer did not foresee.

   It is recommended to use any fixed compiler version from 0.6.12.

   Auditors Remarks: Fixed

   Except for TokenYFIAG.sol, the compiler version is fixed.
   TokenYFIAG.sol - has a compiler version in a range.

2. **Coding Style Issues**

   Coding style issues influence code readability and, in some cases, may lead to bugs in future. Smart Contracts have a naming convention, indentation and code layout issues. It's recommended to use Solidity Style Guide to fix all the issues. Consider following the Solidity guidelines on formatting the code and commenting for all the files. It can improve the overall code quality and readability.

```
Governance.sol
  226:2  error  Line length must be no more than 120 but current length is 133  max-line-length
  279:2  error  Line length must be no more than 120 but current length is 127  max-line-length
  288:2  error  Line length must be no more than 120 but current length is 143  max-line-length

TokenYFIAG.sol
  233:2  error  Line length must be no more than 120 but current length is 132  max-line-length
  421:2  error  Line length must be no more than 120 but current length is 130  max-line-length
  457:2  error  Line length must be no more than 120 but current length is 138  max-line-length

VotingPowerFees.sol
  15:2  error  Line length must be no more than 120 but current length is 124  max-line-length
  80:2  error  Line length must be no more than 120 but current length is 227  max-line-length

VotingPowerFeesAndRewards.sol
  9:2  error  Line length must be no more than 120 but current length is 126  max-line-length

✖ 9 problems (9 errors, 0 warnings)
```

Auditors Remarks: Not Fixed

3. Order of layout

The order of functions as well as the rest of the code layout does not follow the solidity style guide.
Layout contract elements in the following order:

- Pragma statements
- Import statements
- Interfaces
- Libraries
- Contracts

Inside each contract, library or interface, use the following order:

- Type declarations
- State variables
- Events
- Functions

Please read the following documentation links to understand the correct order:

https://solidity.readthedocs.io/en/v0.6.12/style-guide.html#order-of-layout

Auditors Remarks: Not Fixed

4. Naming convention issues – Variables, Structs, Functions

It is recommended to follow all solidity naming conventions to maintain the readability of code.
Details: https://docs.soliditylang.org/en/v0.6.12/style-guide.html#naming-conventions

Auditors Remarks: Not Fixed

5. Remove the dead code from the files

Consider removing the commented or dead code from the files.
Code Lines:
- yWETH.sol: 5-9

Auditors Remarks: Fixed

6. **Log the important events inside functions**

   It’s a good practice to log important events. In Governance.sol for functions like setGovernance, setMinimum and setPeriod should emit events.
   Code Lines: Governance.sol[208, 214, 220]

   Auditors Remarks: Fixed

7. **Use external function modifier instead of public**

   The public functions that are never called by contract should be declared external to save gas. Following functions can be declared external:
   TokenToVotePowerStaking.sol
   - totalSupply() [#36-38]
   - balanceOf(address) [#43-45]
   - stake(uint256) [#50-54]
   - withdraw(uint256) [#58-62]

   Auditors Remarks: Not Fixed

Informational

1. **Overpowered owner**

   The contract is tightly coupled to the owner, making some functions callable only by the owner’s address. This poses a serious risk: if the private key of the owner gets compromised, then an attacker can gain control over the contract.

   Auditors Remarks: Not Fixed

2. **Approve function of ERC-20 is vulnerable**

A security issue called “Multiple Withdrawal Attack” - originates from two methods in the ERC20 standard for approving and transferring tokens. The use of these functions in an adverse environment (e.g., front-running) could result in more tokens being spent than what was intended. This issue is still open on the GitHub, and several solutions have been made to mitigate it.
For more details on how to resolve the multiple withdrawal attack, check the following document for details:
https://drive.google.com/file/d/1skR4BpZ0VBSQICIC_eqRBgGnACf2li5X/view?usp=sharing

Auditors Remarks: Not Fixed

3. **Use of block.timestamp should be avoided**

Do not use block.timestamp, now or blockhash as a source of randomness. Malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. It is risky to use block.timestamp, as block.timestamp can be manipulated by
miners. Therefore block.timestamp is recommended to be supplemented with some other strategy in the case of high-value/risk applications.
Remediations:

- https://consensys.github.io/smart-contract-best-practices/recommendations/#timestamp-dependence
- https://consensys.github.io/smart-contract-best-practices/recommendations/#avoid-using-blocknumber-as-a-timestamp

Auditors Remarks: Not Fixed

Automated Testing

Slither

Slither is an open-source Solidity static analysis framework. This tool provides rich information about Ethereum smart contracts and has the critical properties. It runs a suite of vulnerability detectors, prints visual information about contract details, and provides an API to easily write custom analyses. Slither enables developers to find vulnerabilities, enhance their code comprehension, and quickly prototype custom analyses.

```
Compiled with solc
Number of lines: 528 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 5 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 3
Number of informational issues: 3
Number of low issues: 3
Number of medium issues: 1
Number of high issues: 0

ERCs: ERC20

+------------+--------------+-------+----------------------+--------------+--------------+
|    Name    | # functions  | ERCS  |     ERC20 info       | Complex code |   Features   |
+------------+--------------+-------+----------------------+--------------+--------------+
|  SafeMath  |      8       |       |                      |      No      |              |
| TokenYFIAG |      31      | ERC20 |     No Minting       |      No      | Receive ETH  |
|            |              |       | Approve Race Cond.   |              |              |
|            |              |       |                      |              |              |
+------------+--------------+-------+----------------------+--------------+--------------+
INFO:Slither:TokenYFIAG.sol analyzed (5 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 946 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 9 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 43
Number of low issues: 5
Number of medium issues: 4
Number of high issues: 0

ERCs: ERC20

+---------------+-------------+-------+---------------------+--------------+---------------------+
|      Name     | # functions | ERCS  |     ERC20 info      | Complex code |      Features       |
+---------------+-------------+-------+---------------------+--------------+---------------------+
|    SafeMath   |      13     |       |                     |      No      |                     |
|     IERC20    |      6      | ERC20 |     No Minting      |      No      |                     |
|               |             |       | Approve Race Cond.  |              |                     |
|               |             |       |                     |              |                     |
|   SafeERC20   |      6      |       |                     |      No      |      Send ETH       |
|               |             |       |                     |              | Tokens interaction  |
|    Address    |      11     |       |                     |      No      |      Send ETH       |
|               |             |       |                     |              |   Delegatecall      |
|               |             |       |                     |              |     Assembly        |
|   Controller  |      31     |       |                     |      No      |      Send ETH       |
|               |             |       |                     |              | Tokens interaction  |
| IOneSplitAudit|      2      |       |                     |      No      |    Receive ETH      |
|   IStrategy   |      9      |       |                     |      No      |                     |
+---------------+-------------+-------+---------------------+--------------+---------------------+
INFO:Slither:Controller.sol analyzed (9 contracts)
```

```
Compiled with solc
Number of lines: 1638 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 14 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 13
Number of informational issues: 36
Number of low issues: 15
Number of medium issues: 3
Number of high issues: 0

ERCs: ERC20

+-------------+-------------+-------+---------------------+--------------+---------------------+
|     Name    | # functions | ERCS  |     ERC20 info      | Complex code |      Features       |
+-------------+-------------+-------+---------------------+--------------+---------------------+
|     Math    |      3      |       |                     |      No      |                     |
|   SafeMath  |      13     |       |                     |      No      |                     |
|    ERC20    |      26     | ERC20 |     No Minting      |      No      |                     |
|             |             |       | Approve Race Cond.  |              |                     |
|             |             |       |                     |              |                     |
|  SafeERC20  |      6      |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              | Tokens interaction  |
|   Address   |      11     |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              |   Delegatecall      |
|             |             |       |                     |              |     Assembly        |
|  Governance |      59     |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              | Tokens interaction  |
|             |             |       |                     |              |   AbiEncoderV2      |
| IGovernance |      7      |       |                     |      No      |                     |
+-------------+-------------+-------+---------------------+--------------+---------------------+
INFO:Slither:Governance.sol analyzed (14 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 111 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 3 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 2
Number of informational issues: 6
Number of low issues: 1
Number of medium issues: 0
Number of high issues: 0


+--------------------------------+-------------+------+------------+--------------+----------+
|              Name              | # functions | ERCS | ERC20 info | Complex code | Features |
+--------------------------------+-------------+------+------------+--------------+----------+
| IRewardDistributionRecipient   |      8      |      |            |      No      |          |
+--------------------------------+-------------+------+------------+--------------+----------+
INFO:Slither:IRewardDistributionRecipient.sol analyzed (3 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 1323 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 12 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 15
Number of informational issues: 20
Number of low issues: 4
Number of medium issues: 1
Number of high issues: 0

ERCs: ERC20

+----------------------------+-------------+-------+---------------------+--------------+---------------------+
|            Name            | # functions | ERCS  |     ERC20 info      | Complex code |      Features       |
+----------------------------+-------------+-------+---------------------+--------------+---------------------+
|            Math            |      3      |       |                     |      No      |                     |
|          SafeMath          |      13     |       |                     |      No      |                     |
|           ERC20            |      26     | ERC20 |     No Minting      |      No      |                     |
|                            |             |       | Approve Race Cond.  |              |                     |
|                            |             |       |                     |              |                     |
|         SafeERC20          |      6      |       |                     |      No      |      Send ETH       |
|                            |             |       |                     |              | Tokens interaction  |
|          Address           |      11     |       |                     |      No      |      Send ETH       |
|                            |             |       |                     |              |    Delegatecall     |
|                            |             |       |                     |              |      Assembly       |
| VotingPowerFeesAndRewards  |      38     |       |                     |      No      | Tokens interaction  |
+----------------------------+-------------+-------+---------------------+--------------+---------------------+
INFO:Slither:VotingPowerFeesAndRewards.sol analyzed (12 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 1165 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 11 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 15
Number of informational issues: 19
Number of low issues: 1
Number of medium issues: 1
Number of high issues: 0

ERCs: ERC20

+-------------------------------+-------------+-------+---------------------+--------------+---------------------+
|             Name              | # functions | ERCS  |     ERC20 info      | Complex code |      Features       |
+-------------------------------+-------------+-------+---------------------+--------------+---------------------+
|             Math              |      3      |       |                     |      No      |                     |
|           SafeMath            |      13     |       |                     |      No      |                     |
|            ERC20              |      26     | ERC20 |    No Minting       |      No      |                     |
|                               |             |       | Approve Race Cond.  |              |                     |
|                               |             |       |                     |              |                     |
|          SafeERC20            |      6      |       |                     |      No      |      Send ETH       |
|                               |             |       |                     |              | Tokens interaction  |
|           Address             |      11     |       |                     |      No      |      Send ETH       |
|                               |             |       |                     |              |   Delegatecall      |
|                               |             |       |                     |              |     Assembly        |
| IRewardDistributionRecipient  |      8      |       |                     |      No      |                     |
|       VotingPowerFees         |      15     |       |                     |      No      |      Send ETH       |
|                               |             |       |                     |              | Tokens interaction  |
+-------------------------------+-------------+-------+---------------------+--------------+---------------------+
INFO:Slither:VotingPowerFees.sol analyzed (11 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 12 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 1 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 0
Number of low issues: 0
Number of medium issues: 0
Number of high issues: 0


+-------+-------------+------+------------+--------------+--------------+
|  Name | # functions | ERCS | ERC20 info | Complex code |   Features   |
+-------+-------------+------+------------+--------------+--------------+
| IWETH |      2      |      |            |      No      | Receive ETH |
+-------+-------------+------+------------+--------------+--------------+
INFO:Slither:IWETH.sol analyzed (1 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 12 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 1 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 0
Number of informational issues: 0
Number of low issues: 0
Number of medium issues: 0
Number of high issues: 0


+------+-------------+------+------------+--------------+--------------+
| Name | # functions | ERCS | ERC20 info | Complex code |   Features   |
+------+-------------+------+------------+--------------+--------------+
| WETH |      2      |      |            |      No      | Receive ETH |
+------+-------------+------+------------+--------------+--------------+
INFO:Slither:WETH.sol analyzed (1 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 618 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 5 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 4
Number of informational issues: 11
Number of low issues: 0
Number of medium issues: 0
Number of high issues: 0

ERCs: ERC20

+--------------------------+-------------+-------+----------------------+--------------+----------------------+
|           Name           | # functions | ERCS  |     ERC20 info       | Complex code |      Features        |
+--------------------------+-------------+-------+----------------------+--------------+----------------------+
|         SafeMath         |      13     |       |                      |      No      |                      |
|          IERC20          |      6      | ERC20 |     No Minting       |      No      |                      |
|                          |             |       | Approve Race Cond.   |              |                      |
|                          |             |       |                      |              |                      |
|        SafeERC20         |      6      |       |                      |      No      |      Send ETH        |
|                          |             |       |                      |              | Tokens interaction   |
|         Address          |      11     |       |                      |      No      |      Send ETH        |
|                          |             |       |                      |              |    Delegatecall      |
|                          |             |       |                      |              |      Assembly        |
| TokenToVotePowerStaking  |      6      |       |                      |      No      |                      |
+--------------------------+-------------+-------+----------------------+--------------+----------------------+
INFO:Slither:TokenToVotePowerStaking.sol analyzed (5 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 1229 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 10 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 9
Number of informational issues: 21
Number of low issues: 6
Number of medium issues: 1
Number of high issues: 0

ERCs: ERC20

+-------------+-------------+-------+---------------------+--------------+---------------------+
|     Name    | # functions |  ERCS |     ERC20 info      | Complex code |      Features       |
+-------------+-------------+-------+---------------------+--------------+---------------------+
|   SafeMath  |      13     |       |                     |      No      |                     |
|  SafeERC20  |      6      |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              | Tokens interaction  |
|   Address   |      11     |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              |   Delegatecall      |
|             |             |       |                     |              |      Assembly       |
| IController |      8      |       |                     |      No      |                     |
|    yVault   |      55     | ERC20 |      Pausable       |      No      |      Send ETH       |
|             |             |       |     No Minting      |              | Tokens interaction  |
|             |             |       | Approve Race Cond.  |              |                     |
|             |             |       |                     |              |                     |
+-------------+-------------+-------+---------------------+--------------+---------------------+
INFO:Slither:yVault.sol analyzed (10 contracts)
```

```
INFO:Printers:
Compiled with solc
Number of lines: 1314 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 12 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 9
Number of informational issues: 24
Number of low issues: 6
Number of medium issues: 2
Number of high issues: 1
ERCs: ERC20

+-------------+-------------+-------+---------------------+--------------+---------------------+
|     Name    | # functions |  ERCS |     ERC20 info      | Complex code |      Features       |
+-------------+-------------+-------+---------------------+--------------+---------------------+
|   SafeMath  |      13     |       |                     |      No      |                     |
|  SafeERC20  |      6      |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              | Tokens interaction  |
|   Address   |      11     |       |                     |      No      |      Send ETH       |
|             |             |       |                     |              |   Delegatecall      |
|             |             |       |                     |              |      Assembly       |
| IController |      8      |       |                     |      No      |                     |
|    IWETH    |      2      |       |                     |      No      |    Receive ETH      |
|    yWETH    |      60     | ERC20 |      Pausable       |      No      |    Receive ETH      |
|             |             |       |     No Minting      |              |      Send ETH       |
|             |             |       | Approve Race Cond.  |              | Tokens interaction  |
|             |             |       |                     |              |                     |
+-------------+-------------+-------+---------------------+--------------+---------------------+
INFO:Slither:yWETH.sol analyzed (12 contracts)
```

Slither didn't raise any critical issue with smart contracts. The smart contracts were well tested, and all the minor issues that were raised have been documented in the report. Also, all other vulnerabilities of importance have already been covered in the manual audit section of the report.

SmartCheck

SmartCheck is a tool for automated static analysis of Solidity source code for security vulnerabilities and best practices. SmartCheck translates Solidity source code into an XML-based intermediate representation and checks it against XPath patterns. SmartCheck shows significant improvements over existing alternatives in terms of false discovery rate (FDR) and false negative rate (FNR).

```
ruleId: SOLIDITY_VISIBILITY
patternId: b51ce0
severity: 1
line: 325
column: 14
content: ();

SOLIDITY_VISIBILITY :2
SOLIDITY_SAFEMATH :1
SOLIDITY_PRAGMAS_VERSION :2
SOLIDITY_PRIVATE_MODIFIER_DONT_HIDE_DATA :7
SOLIDITY_ADDRESS_HARDCODED :1
```

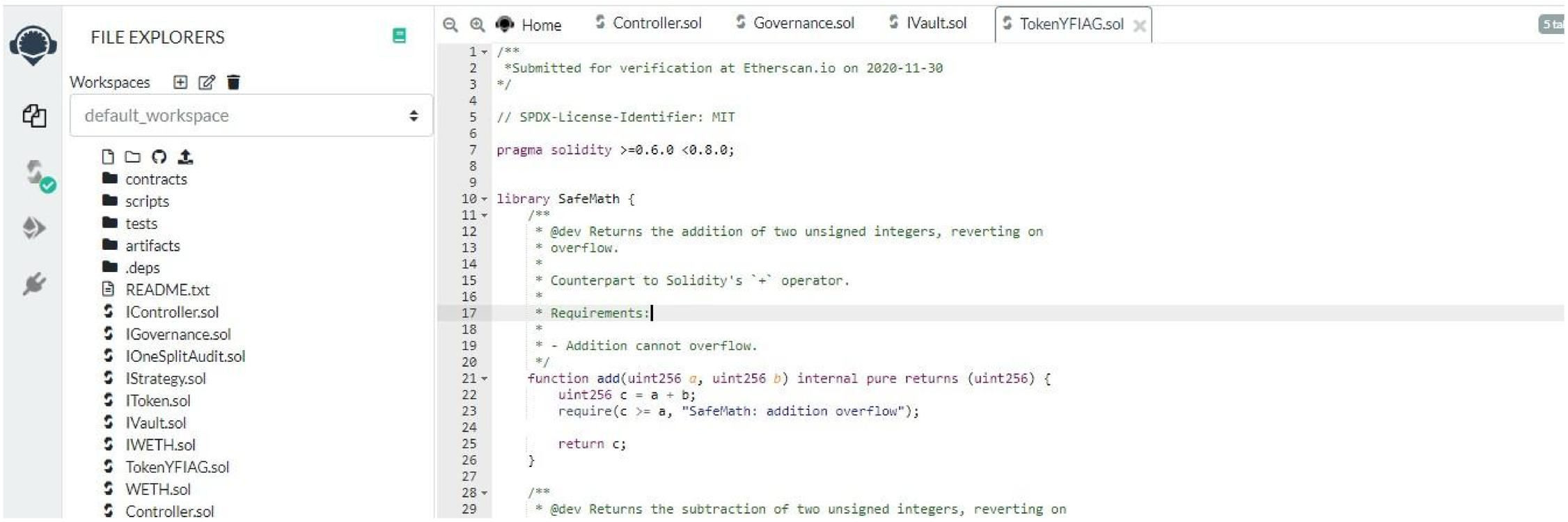
SmartCheck did not detect any high severity issue. All the considerable issues raised by SmartCheck are already covered in the code review section of this report.

Mythril

Mythril is a security analysis tool for EVM bytecode. It detects security vulnerabilities in smart contracts built for Ethereum. It uses symbolic execution, SMT solving and taint analysis to detect a variety of security vulnerabilities.
Mythril did not detect any high severity issue. All the considerable issues raised by Mythril are already covered in the issues found section of this report.

Remix IDE

Remix is a powerful, open-source tool that helps you write Solidity contracts straight from the browser. Written in JavaScript, Remix supports both usage in the browser and locally.



The smart contract compiled without any error on Remix IDE. **The contract was successfully deployed. And the gas consumption was found to be normal.**

Closing Summary

Overall, the smart contract code is extremely well documented, follows a high-quality software development standard, contains many utilities and automation scripts to support continuous deployment/ testing/ integration, and does NOT contain any obvious exploitation vectors that QuillAudits was able to leverage within the timeframe of testing allotted. Overall, the smart contracts adhered to ERC20 guidelines. No critical or major vulnerabilities were found in the audit. **One issue of medium severity and several issues of low severity were found and reported during the audit**. A few of them are fixed now.

The outcome of this security audit is satisfactory; due to time and resource constraints, only testing and verification of essential properties was performed to achieve objectives and deliverables set in the scope. QuillAudits recommends performing further testing to validate extended safety and correctness in context to the whole set of contracts.

Disclaimer

QuillHash audit is not a security warranty, investment advice, or an endorsement of the YearnAgnostic platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the YearnAgnostic Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

YearnAgnostic

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com