HALBORN

Irrigation - Protocol

Smart Contract Security Assessment

Prepared by: **Halborn**

Date of Engagement: **June 23rd, 2023 - August 4th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE
0.1	Document Creation	06/23/2023
0.2	Document Updates	06/23/2023
0.3	Document Updates	08/04/2023
0.4	Draft Review	08/04/2023
0.5	Draft Review	08/04/2023
1.0	Remediation Plan	09/25/2023
1.1	Remediation Plan Review	09/25/2023
1.2	Remediation Plan Review	09/25/2023

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Gokberk Gulgun	Halborn	Gokberk.Gulgun@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Irrigation Protocol engaged Halborn to conduct a security assessment on their smart contracts beginning on June 23rd, 2023 and ending on August 4th, 2023. The security assessment was scoped to the smart contracts provided in the following GitHub repositories:

- IrrigationProtocol/irrigation-contracts-diamond.

1.2 ASSESSMENT SUMMARY

The team at Halborn was provided six weeks for the engagement and assigned a full-time security engineer to verify the security of the smart contracts. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this assessment is to:

- Ensure that smart contract functions operate as intended.
- Identify potential security issues with the smart contracts.

In summary, Halborn identified some security risks that were mostly addressed by the Irrigation Protocol team.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this assessment. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the code and can quickly identify items that do not follow the security best practices. The following phases and associated tools were used during the assessment:

- Research into architecture and purpose.
- Smart contract manual code review and walkthrough.
- Graphing out functionality and contract logic/connectivity/functions. (solgraph)
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes.
- Manual testing by custom scripts.
- Scanning of solidity files for vulnerabilities, security hot-spots or bugs. (MythX)
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Testnet deployment. (Foundry)

2. RISK METHODOLOGY

Every vulnerability and issue observed by Halborn is ranked based on **two sets** of **Metrics** and a **Severity Coefficient**. This system is inspired by the industry standard Common Vulnerability Scoring System.

The two **Metric sets** are: **Exploitability** and **Impact**. **Exploitability** captures the ease and technical means by which vulnerabilities can be exploited and **Impact** describes the consequences of a successful exploit.

The **Severity Coefficients** is designed to further refine the accuracy of the ranking with two factors: **Reversibility** and **Scope**. These capture the impact of the vulnerability on the environment as well as the number of users and smart contracts affected.

The final score is a value between 0-10 rounded up to 1 decimal place and 10 corresponding to the highest security risk. This provides an objective and accurate rating of the severity of security vulnerabilities in smart contracts.

The system is designed to assist in identifying and prioritizing vulnerabilities based on their level of risk to address the most critical issues in a timely manner.

2.1 EXPLOITABILITY

Attack Origin (AO):

Captures whether the attack requires compromising a specific account.

Attack Cost (AC):

Captures the cost of exploiting the vulnerability incurred by the attacker relative to sending a single transaction on the relevant blockchain. Includes but is not limited to financial and computational cost.

Attack Complexity (AX):

Describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. Includes but is not limited to macro situation, available third-party liquidity and regulatory challenges.

Metrics:

Exploitability Metric (m_E)	Metric Value	Numerical Value
Attack Origin (AO)	Arbitrary (AO:A) Specific (AO:S)	1 0.2
Attack Cost (AC)	Low (AC:L) Medium (AC:M) High (AC:H)	1 0.67 0.33
Attack Complexity (AX)	Low (AX:L) Medium (AX:M) High (AX:H)	1 0.67 0.33

Exploitability E is calculated using the following formula:

$$E = \prod m_e$$

2.2 IMPACT

Confidentiality (C):

Measures the impact to the confidentiality of the information resources managed by the contract due to a successfully exploited vulnerability. Confidentiality refers to limiting access to authorized users only.

Integrity (I):

Measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of data stored and/or processed on-chain. Integrity impact directly affecting Deposit or Yield records is excluded.

Availability (A):

Measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. This metric refers to smart contract features and functionality, not state. Availability impact directly affecting Deposit or Yield is excluded.

Deposit (D):

Measures the impact to the deposits made to the contract by either users or owners.

Yield (Y):

Measures the impact to the yield generated by the contract for either users or owners.

Metrics:

Impact Metric (m_I)	Metric Value	Numerical Value
Confidentiality (C)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Integrity (I)	None (I:N)	0
	Low (I:L)	0.25
	Medium (I:M)	0.5
	High (I:H)	0.75
	Critical (I:C)	1
Availability (A)	None (A:N)	0
	Low (A:L)	0.25
	Medium (A:M)	0.5
	High (A:H)	0.75
	Critical	1
Deposit (D)	None (D:N)	0
	Low (D:L)	0.25
	Medium (D:M)	0.5
	High (D:H)	0.75
	Critical (D:C)	1
Yield (Y)	None (Y:N)	0
	Low (Y:L)	0.25
	Medium: (Y:M)	0.5
	High: (Y:H)	0.75
	Critical (Y:H)	1

Impact I is calculated using the following formula:

$$I = max(m_I) + \frac{\sum m_I - max(m_I)}{4}$$

2.3 SEVERITY COEFFICIENT

Reversibility (R):

Describes the share of the exploited vulnerability effects that can be reversed. For upgradeable contracts, assume the contract private key is available.

Scope (S):

Captures whether a vulnerability in one vulnerable contract impacts resources in other contracts.

Coefficient (C)	Coefficient Value	Numerical Value
Reversibility (r)	None (R:N) Partial (R:P) Full (R:F)	1 0.5 0.25
Scope (s)	Changed (S:C) Unchanged (S:U)	1.25 1

Severity Coefficient C is obtained by the following product:

$$C = rs$$

The Vulnerability Severity Score S is obtained by:

$$S = min(10, EIC * 10)$$

The score is rounded up to 1 decimal places.

Severity	Score Value Range
Critical	9 - 10
High	7 - 8.9
Medium	4.5 - 6.9
Low	2 - 4.4
Informational	0 - 1.9

2.4 SCOPE

1. IN-SCOPE TREE & COMMIT :

The security assessment was scoped to the following smart contracts:

GitHub repository: IrrigationProtocol/irrigation-contracts-diamond
Commit ID: 159927ff1e8f8e212cd09894c29c1019e5d417f6
Remediation Commit ID: 1b9420fb77d856bb57a35e947ae255b035e3037c
Smart contracts in scope:

- SprinklerUpgradeable.sol
- WaterTowerUpgradeable.sol
- TrancheBondUpgradeable.sol
- AuctionUpgradeable.sol
- ERC1155WhitelistUpgradeable.sol
- WaterCommonUpgradeable.sol
- PriceOracleUpgradeable.sol
- PodsOracleUpgradeable.sol
- WaterUpgradeable.sol
- SprinklerStorage.sol
- WaterTowerStorage.sol
- TrancheBondStorage.sol
- AuctionStorage.sol
- ERC1155WhitelistStorage.sol
- WaterCommonStorage.sol
- PriceOracleStorage.sol
- IrrigationDiamond.sol
- Diamond.sol
- DiamondCutFacet.sol
- DiamondLoupeFacet.sol
- EIP2535Initializable.sol
- IrrigationAccessControl.sol
- OwnershipFacet.sol
- FullMath.sol
- PodTransferHelper.sol
- LibPrice.sol

- BeanPriceOracle.sol
- ChainlinkOracle.sol
- UniswapV3Twap.sol

3. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
2	0	4	1	2

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) SPRINKLERUPGRADEABLE CONTRACT CAN BE DRAINED THROUGH THE EXCHANGETOWATER FUNCTION	Critical (10)	SOLVED - 08/21/2023
(HAL-02) AUCTIONS AND BIDS COULD BE STUCK PERMANENTLY IN THE AUCTIONUPGRADEABLE CONTRACT IF A BID IS PLACED BY A BLACKLISTED USDC/USDT USER	Critical (10)	SOLVED - 08/21/2023
(HAL-03) BIDS CAN BE DOSED BY PLACING A VERY HIGH BID ON A VERY SMALL BIDAMOUNT	Medium (5.0)	SOLVED - 08/21/2023
(HAL-04) LATESTANSWER CALL MAY RETURN STALE RESULTS	Medium (6.2)	SOLVED - 08/21/2023
(HAL-05) USER COULD CANCEL THE REST OF BIDS OF AN AUCTION BY DOING 200 DIFFERENT BIDS	Medium (5.6)	SOLVED - 08/21/2023
(HAL-06) SWAPETHFORWATER CALL CAN BE SANDWICHED	Medium (5.0)	SOLVED - 08/21/2023
(HAL-07) AUTOIRRIGATECALL WILL ALWAYS REVERT WITH OVERFLOW IF ITS CALLED WITH THE FULL REWARDAMOUNT	Low (2.5)	RISK ACCEPTED
(HAL-08) LACK OF A DOUBLE-STEP TRANSFEROWNERSHIP PATTERN	Informational (0.0)	ACKNOWLEDGED
(HAL-09) CALLING SETMIDDLEASSET WOULD CAUSE THE IRRIGATION BONUS TO ALWAYS BE ZERO	Informational (0.0)	ACKNOWLEDGED

FINDINGS & TECH DETAILS

4.1 (HAL-01) SPRINKLERUPGRADEABLE CONTRACT CAN BE DRAINED THROUGH THE EXCHANGETOWATER FUNCTION - CRITICAL(10)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

In the SprinklerUpgradeable contract, a set of assets are whitelisted which can be used to be swapped for WATER. The assets whitelisted can be seen here:
- Native Ether represented by the address: 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE
- USDC
- DAI
- USDT
- LUSD
- BEAN
- ROOT
- SPOT
- OHM
- PAXG
- CNHT

Native Ether, represented by the 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE address, is used to check the multiplier assigned to the Native Ether exchanges:

Listing 1: SprinklerUpgradeable.sol (Line 100)

```
94 /**
95  * @notice Exchange ETH to water
96  * @return waterAmount received water amount
97  */
```

```
98 function exchangeETHToWater() external payable nonReentrant
 ↳ returns (uint256 waterAmount) {
99     require(msg.value != 0, "Invalid amount");
100    waterAmount = getWaterAmount(Constants.ETHER, msg.value);
101    if (waterAmount > sprinkleableWater()) revert
 ↳ InsufficientWater();
102    require(waterAmount != 0, "No water output"); // if price is 0
 ↳  or tokenMultiplier is 0, amount can be 0
103    transferWater(waterAmount);
104    SprinklerStorage.layout().reserves[Constants.ETHER] += msg.
 ↳ value;
105    emit WaterExchanged(msg.sender, Constants.ETHER, msg.value,
 ↳ waterAmount, false);
106 }
```

Listing 2: SprinklerUpgradeable.sol (Line 155)

```
151 function getWaterAmount(
152     address _token,
153     uint256 _amount
154 ) public view returns (uint256 waterAmount) {
155     uint256 multiplier = tokenMultiplier(_token);
156     uint256 tokenPrice = IPriceOracleUpgradeable(address(this)).
 ↳ getPrice(_token);
157     uint256 waterPrice = IPriceOracleUpgradeable(address(this)).
 ↳ getWaterPrice();
158     waterAmount = (_amount * tokenPrice * multiplier) / waterPrice
 ↳ ;
159 }
```

The function exchangeTokenToWater() is used to exchange whitelisted assets for WATER:

Listing 3: SprinklerUpgradeable.sol (Lines 80,88)

```
71 /**
72  * @notice Exchange whitelisted asset(BEAN, BEAN:3CRV, Spot, and
 ↳ so on) to water
73  * @param token source token address
74  * @param amount source token amount
75  * @return waterAmount received water amount
76  */
```

```
77 function exchangeTokenToWater(
78     address token,
79     uint256 amount
80 ) external onlyListedAsset(token) nonReentrant returns (uint256
↳ waterAmount) {
81     require(token != address(this), "Invalid token");
82     require(amount != 0, "Invalid amount");
83 
84     waterAmount = getWaterAmount(token, amount);
85     if (waterAmount > sprinkleableWater()) revert
↳ InsufficientWater();
86     require(waterAmount != 0, "No water output"); // if price is
↳ 0, amount can be 0
87 
88     TransferHelper.safeTransferFrom(token, msg.sender, address(
↳ this), amount);
89     transferWater(waterAmount);
90     SprinklerStorage.layout().reserves[token] += amount;
91     emit WaterExchanged(msg.sender, token, amount, waterAmount,
↳ false);
92 }
```

This function makes use of the TransferHelper library:

Listing 4: TransferHelper.sol (Line 21)

```
18 function safeTransferFrom(address token, address from, address to,
↳  uint value) internal {
19     // bytes4(keccak256(bytes('transferFrom(address,address,
↳ uint256)')));
20     (bool success, bytes memory data) = token.call(abi.
↳ encodeWithSelector(0x23b872dd, from, to, value));
21     require(success && (data.length == 0 || abi.decode(data, (bool
↳ ))), 'STF');
22 }
```

Although, this library, unlike OpenZeppelin's SafeERC20, does not check that the token address is actually a smart contract with some code deployed in that address (&& address(token).code.length > 0):

Listing 5: SafeERC20.sol (Line 14)

```
 1 /**
 2  * @dev Transfer `value` amount of `token` from `from` to `to`,
↳ spending the approval given by `from` to the
 3  * calling contract. If `token` returns no value, non-reverting
↳ calls are assumed to be successful.
 4  */
 5 function safeTransferFrom(IERC20 token, address from, address to,
↳ uint256 value) internal {
 6     _callOptionalReturn(token, abi.encodeCall(token.transferFrom,
↳ (from, to, value)));
 7 }
 8 function _callOptionalReturnBool(IERC20 token, bytes memory data)
↳ private returns (bool) {
 9     // We need to perform a low level call here, to bypass
↳ Solidity's return data size checking mechanism, since
10     // we're implementing it ourselves. We cannot use {Address-
↳ functionCall} here since this should return false
11     // and not revert is the subcall reverts.
12
13     (bool success, bytes memory returndata) = address(token).call(
↳ data);
14     return success && (returndata.length == 0 || abi.decode(
↳ returndata, (bool))) && address(token).code.length > 0;
15 }
```

Based on this and considering that the 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE address is a representative address with no actual code deployed in it, a malicious user could call SprinklerUpgradeable.exchangeTokenToWater("0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE", <very high amount>) and drain all the WATER tokens from the contract.

Proof of Concept:

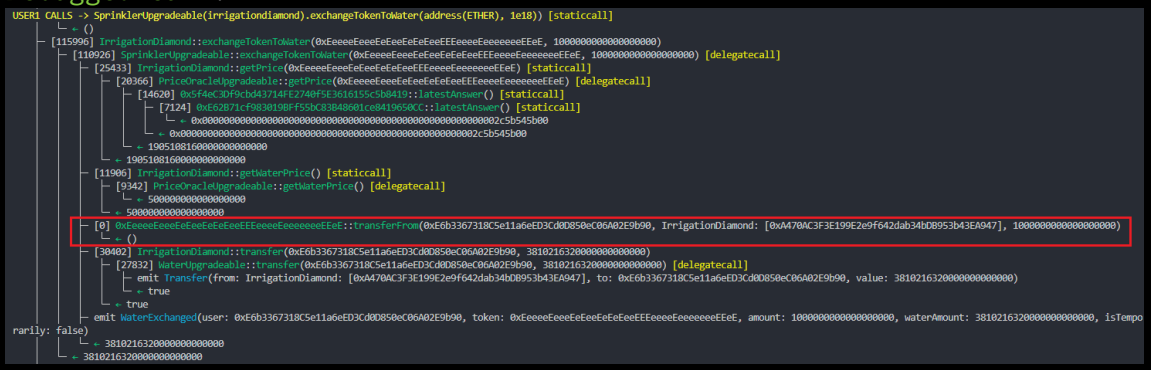
Exploit:

```
test_exchangeEther
  ____________________

  WaterUpgradeable(irrigationdiamond).balanceOf(user1) -> 0

USER1 CALLS -> SprinklerUpgradeable(irrigationdiamond).exchangeTokenToWater(address(ETHER), 1e18)
  WaterUpgradeable(irrigationdiamond).balanceOf(user1) -> 38102163200000000000000
```

Debugged call:



BVSS:

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:C/Y:N/R:N/S:U (10)

Recommendation:

It is recommended to use the OpenZeppelin's SafeERC20 library to perform all the token transfers instead of the TransferHelper library.

Remediation Plan:

SOLVED: The Irrigation Protocol team solved the issue by implementing the recommended solution.

Commit ID : 2a75e858618864b3962c623035cca5b5aef4ff4d.

4.2 (HAL-02) AUCTIONS AND BIDS COULD BE STUCK PERMANENTLY IN THE AUCTIONUPGRADEABLE CONTRACT IF A BID IS PLACED BY A BLACKLISTED USDC/USDT USER - CRITICAL(10)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

The AuctionUpgradeable contract allows creating different auctions through the createAuction() function:

Listing 6: AuctionUpgradeable.sol

```
100 function createAuction(
101     uint96 startTime,
102     uint96 duration,
103     address sellToken,
104     uint256 trancheIndex,
105     uint128 sellAmount,
106     uint128 minBidAmount,
107     uint128 fixedPrice,
108     uint128 priceRangeStart,
109     uint128 priceRangeEnd,
110     AuctionType auctionType
111 ) external payable returns (uint256)
```

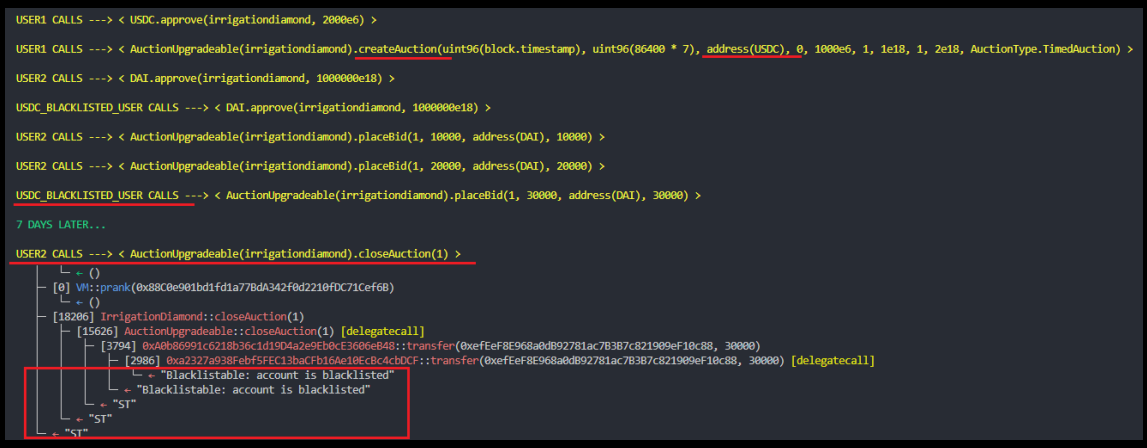
Once an auction is completed, the closeAuction() function will be called and will distribute the sellToken between all the bidders. As currently USDC contains a blacklist of users which cannot transfer or receive this token, the following issue could occur:

1. Alice creates an auction of 1000 USDC(sellToken).
2. Users place different bids for the USDC auction.

3. A Blacklisted USDC user also places a bid for the USDC auction. The bid is placed using a whitelisted purchaseToken like DAI.
4. The auction is completed. closeAuction() is called, but it reverts every time as the USDC transfer to the blacklisted USDC user is forbidden.
5. The auction USDC sellTokens are now stuck in the contract. All the bids are also stuck permanently in the contract.

- Similar issue can occur with USDT as it also contains a blacklist of users.

Proof of Concept:



BVSS:

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:C/Y:N/R:N/S:U (10)

Recommendation:

Consider adding a try/catch code block for the safeTransfer() call in the AuctionUpgradeable._settleBid() function.

Remediation Plan:

SOLVED: The Irrigation Protocol team solved the issue by enforcing a whitelist on the sellTokens. The whitelist will not include blacklistable tokens like USDC/USDT.

Commit ID : cae1d1718ef504f4fc27ba1aa464600a71ae635a.

4.3 (HAL-03) BIDS CAN BE DOSED BY PLACING A VERY HIGH BID ON A VERY SMALL BIDAMOUNT - MEDIUM (5.0)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

As mentioned before, the AuctionUpgradeable contract allows creating different auctions through the createAuction() function:

Listing 7: AuctionUpgradeable.sol (Line 106)

```
100 function createAuction(
101     uint96 startTime,
102     uint96 duration,
103     address sellToken,
104     uint256 trancheIndex,
105     uint128 sellAmount,
106     uint128 minBidAmount,
107     uint128 fixedPrice,
108     uint128 priceRangeStart,
109     uint128 priceRangeEnd,
110     AuctionType auctionType
111 ) external payable returns (uint256)
```

The minBidAmount parameter sets the minimum amount of the token that can be bid. In the case that the auctioneer sets the minBidAmount to a very low value, i.e. 1, the following attack vector would be possible:

1. Alice (auctioneer) creates an auction of 1000 USDC tokens and sets minBidAmount to 1. The auctionType is a TimedAuction.
2. Bob calls placeBid(1, 1, DAI, 1000e18) bidding 0.01 DAI for 0.000001 USDC. (bidPrice = 1000 DAI).
3. Joe calls placeBid(1, 100e6, DAI, 1e18) bidding 100 DAI for 100 USDC. (bidPrice = 1 DAI) but it reverts with a low Bid error.

4. If any user wants to bid now, they should place a bid higher than the abusive bid placed by Bob. (bidPrice = 1000 DAI). Bob managed to cause a Denial of Service, losing only 0.01 DAI.

Proof of Concept:

```
USER1 CALLS ---> < USDC.approve(irrigationdiamond, 2000e6) >

USER1 CALLS ---> < AuctionUpgradeable(irrigationdiamond).createAuction(uint96(block.timestamp), uint96(86400 * 7), address(USDC), 0, 1000e6, 1, 1e18, 1e16, 2e18, AuctionType.TimedAuction) >

USER2 CALLS ---> < DAI.approve(irrigationdiamond, 1000e18) >
  DAI.balanceOf(user2) -> 10000000000000000000
  USDC.balanceOf(user2) -> 0

USER2 CALLS ---> < AuctionUpgradeable(irrigationdiamond).placeBid(1, 1, address(DAI), 1000e18) >
  DAI.balanceOf(user2) -> 9999990000000000000
  USDC.balanceOf(user2) -> 0

USER3 CALLS ---> < DAI.approve(irrigationdiamond, 10000000e18) >
  DAI.balanceOf(user3) -> 10000000000000000000000000
  USDC.balanceOf(user3) -> 0

USER3 CALLS ---> < AuctionUpgradeable(irrigationdiamond).placeBid(1, 100e6, address(DAI), 1e18) >) [staticcall]
    │   └─ ← ()
    ├─ [0] VM::prank(0x7231C364597f3BfDB72Cf52b197cc59111e71794)
    │   └─ ← ()
    ├─ [5036] IrrigationDiamond::placeBid(1, 100000000, 0x6B175474E89094C44Da98b954EedeAC495271d0F, 100000000000000000)
    │   ├─ [4444] AuctionUpgradeable::placeBid(1, 100000000, 0x6B175474E89094C44Da98b954EedeAC495271d0F, 100000000000000000) [delegatecall]
    │   │   └─ ← "low Bid"
    │   └─ ← "low Bid"
    └─ ← "low Bid"

Test result: FAILED. 0 passed; 1 failed; finished in 27.20s

Failing tests:
Encountered 1 failing test in test/foundry/AuctionUpgradeable.t.sol:AuctionUpgradeableTests
[FAIL. Reason: low Bid] test_HAL02() (gas: 1039988)
```

BVSS:

AO:A/AC:L/AX:L/C:N/I:M/A:N/D:N/Y:N/R:N/S:U (5.0)

Recommendation:

It is recommended to set a minimum minBidAmount threshold of, for example 1% of the total sellAmount.

Remediation Plan:

SOLVED: The Irrigation Protocol team solved the issue by implementing the recommended solution.

Commit ID : 0a94a93c0f191c992e9f75a41d630ae3dbf80de8.

4.4 (HAL-04) LATESTANSWER CALL MAY RETURN STALE RESULTS - MEDIUM (6.2)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

In the ChainlinkOracle contract, the function getChainlinkPrice() is used to retrieve prices from different Chainlink aggregators:

Listing 8: ChainlinkOracle.sol (Lines 35,37)

```
29 /// @dev returns price with decimals 18
30 function getChainlinkPrice(AggregatorV2V3Interface feed) internal
↳ view returns (uint256) {
31     // Chainlink USD-denominated feeds store answers at 8 decimals
32     uint256 decimalDelta = uint256(18) - feed.decimals();
33     // Ensure that we don't multiply the result by 0
34     if (decimalDelta > 0) {
35         return uint256(feed.latestAnswer()) * 10 ** decimalDelta;
36     } else {
37         return uint256(feed.latestAnswer());
38     }
39 }
```

According to the Chainlink's documentation this function is deprecated and should not be used. Moreover, using latestAnswer() could lead to return invalid/stale prices.

BVSS:

AO:A/AC:L/AX:L/C:N/I:M/A:N/D:N/Y:N/R:N/S:C (6.2)

Recommendation:

It is recommended to update the getChainlinkPrice() function as shown below:

Listing 9: getChainlinkPrice() new (Lines 1,8,9,10,11,14,15,16,17)

```
1  uint256 private constant GRACE_PERIOD_TIME = 3600;
2  /// @dev returns price with decimals 18
3  function getChainlinkPrice(AggregatorV2V3Interface feed) internal
↳ view returns (uint256) {
4      // Chainlink USD-denominated feeds store answers at 8 decimals
5      uint256 decimalDelta = uint256(18) - feed.decimals();
6      // Ensure that we don't multiply the result by 0
7      if (decimalDelta > 0) {
8          (uint80 roundId, int256 price, uint startedAt, uint
↳ updatedAt, uint80 answeredInRound) = feed.latestRoundData();
9          require(answeredInRound >= roundID, "Stale price");
10         require(price > 0, "invalid price");
11         require(block.timestamp <= updatedAt + GRACE_PERIOD_TIME,
↳ "Stale price");
12         return uint256(price) * 10 ** decimalDelta;
13     } else {
14         (uint80 roundId, int256 price, uint startedAt, uint
↳ updatedAt, uint80 answeredInRound) = feed.latestRoundData();
15         require(answeredInRound >= roundID, "Stale price");
16         require(price > 0, "invalid price");
17         require(block.timestamp <= updatedAt + GRACE_PERIOD_TIME,
↳ "Stale price");
18         return uint256(price);
19     }
20 }
```

Remediation Plan:

SOLVED: The Irrigation Protocol team solved the issue by implementing the recommended solution.

Commit ID : 57d3945089a52425b91cdc8005b62b73d5648c68.

4.5 (HAL-05) USER COULD CANCEL THE REST OF BIDS OF AN AUCTION BY DOING 200 DIFFERENT BIDS - MEDIUM (5.6)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

In the AuctionUpgradeable contract, the internal function _settleAuction() is called every time an auction is closed:

Listing 10: AuctionUpgradeable.sol (Line 84)

```
310 function _settleAuction(uint256 auctionId) internal {
311     AuctionData memory auction = AuctionStorage.layout().auctions[
 ↳ auctionId];
312     uint256 trancheIndex = auction.trancheIndex;
313     // when there are no bids, all token amount will be transfered
 ↳  back to seller
314     if (auction.curBidId == 0) {
315         if (trancheIndex == 0) {
316             TransferHelper.safeTransfer(auction.sellToken, auction
 ↳ .seller, auction.reserve);
317         } else {
318             IERC1155Upgradeable(address(this)).safeTransferFrom(
319                 address(this),
320                 auction.seller,
321                 trancheIndex,
322                 auction.reserve,
323                 Constants.EMPTY
324             );
325         }
326 
327         AuctionStorage.layout().auctions[auctionId].reserve = 0;
328         AuctionStorage.layout().auctions[auctionId].status =
 ↳ AuctionStatus.Closed;
329         emit AuctionClosed(auction.reserve, auctionId);
330         return;
331     }
```

```
332     uint128 availableAmount = auction.reserve;
333     uint256 settledBidCount = 0;
334     uint256 curBidId = auction.curBidId;
335     do {
336         Bid memory bid = AuctionStorage.layout().bids[auctionId][
 ↳ curBidId];
337         uint128 settledAmount = _settleBid(
338             auction.sellToken,
339             auction.trancheIndex,
340             auction.seller,
341             bid,
342             availableAmount
343         );
344         availableAmount -= settledAmount;
345         AuctionStorage.layout().bids[auctionId][curBidId].bCleared
 ↳  = true;
346         --curBidId;
347         ++settledBidCount;
348     } while (
349         curBidId > 0 &&
350             settledBidCount <= MAX_CHECK_BID_COUNT &&
351             availableAmount >= auction.minBidAmount
352     );
353     if (availableAmount > 0) {
354         if (auction.assetType == AssetType.ERC20) {
355             TransferHelper.safeTransfer(auction.sellToken, auction
 ↳ .seller, availableAmount);
356         } else {
357             IERC1155Upgradeable(address(this)).safeTransferFrom(
358                 address(this),
359                 auction.seller,
360                 trancheIndex,
361                 availableAmount,
362                 Constants.EMPTY
363             );
364         }
365     }
366
367     AuctionStorage.layout().auctions[auctionId].reserve = 0;
368     AuctionStorage.layout().auctions[auctionId].status =
 ↳ AuctionStatus.Closed;
369     emit AuctionClosed(availableAmount, auctionId);
370 }
```

This function iterates over all the bids, starting with the last one, transferring the purchased tokens to the bidder until reaching MAX_CHECK_BID_COUNT (which is hardcoded to 200). The rest of the bids will be automatically cancelled.

Based on this, the following attack vector could be possible:
1. Alice creates an auction:

Listing 11

```
AuctionUpgradeable(irrigationdiamond).createAuction(
    uint96(block.timestamp),
    uint96(86400 * 7),
    address(USDC),
    0,
    1000e6,
    1,
    1e18,
    1,
    2e18,
    AuctionType.TimedAuction
);
```

2. Different users place different bids. In total, 150 bids were placed.
3. Alice is not happy with the current bid amounts, so she places 200 different bids on her own auction, 1 second before the auction is completed.
4. The auction is completed. Alice gets her 200 bids, although the rest of the bids of the other users placed before would be automatically canceled.

BVSS:

AO:A/AC:L/AX:L/C:N/I:M/A:N/D:L/Y:N/R:N/S:U (5.6)

Recommendation:

Consider redesigning the logic of the `_settleAuction()` function, so such large number of iterations are not required.

Remediation Plan:

SOLVED: The Irrigation Protocol team solved the issue by implementing the recommended solution.

Commit ID : c269697ae4b7997abec4c836d46e3f73f292af03.

4.6 (HAL-06) SWAPETHFORWATER CALL CAN BE SANDWICHED - MEDIUM (5.0)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

In the WaterTowerUpgradeable contract, the function _swapEthForWater() is used to swap Ether for Beans using the Curve Router:

Listing 12: WaterTowerUpgradeable.sol (Lines 200-202)

```
180 function _swapEthForWater(uint256 amount) internal returns (
 ↳ uint256 waterAmount) {
181     if (WaterTowerStorage.layout().middleAssetForIrrigate ==
 ↳ Constants.BEAN) {
182         /// @dev swap ETH for BEAN using curve router
183         address[9] memory route = [
184             Constants.ETHER,
185             Constants.TRI_CRYPTO_POOL,
186             Constants.USDT,
187             Constants.CURVE_BEAN_METAPOOL,
188             Constants.BEAN,
189             Constants.ZERO,
190             Constants.ZERO,
191             Constants.ZERO,
192             Constants.ZERO
193         ];
194         uint256[3][4] memory swapParams = [
195             [uint(2), 0, 3],
196             [uint(3), 0, 2],
197             [uint(0), 0, 0],
198             [uint(0), 0, 0]
199         ];
200         uint256 beanAmount = ICurveSwapRouter(Constants.
 ↳ CURVE_ROUTER).exchange_multiple{
201             value: amount
202         }(route, swapParams, amount, 0);
203
```

```
204         waterAmount = ISprinklerUpgradeable(address(this)).
 ↳ getWaterAmount(
205             Constants.BEAN,
206             beanAmount
207         );
208     }
209 }
```

This function is calling the exchange_multiple function with no prevention against slippage:

Listing 13: CURVE_ROUTER contract (Line 7)

```
1 @external
2 @payable
3 def exchange_multiple(
4     _route: address[9],
5     _swap_params: uint256[3][4],
6     _amount: uint256,
7     _expected: uint256,
8     _pools: address[4]=[ZERO_ADDRESS, ZERO_ADDRESS, ZERO_ADDRESS,
 ↳ ZERO_ADDRESS],
9     _receiver: address=msg.sender
10 ) -> uint256:
```

Based on this, any autoIrrigate() or irrigate() call could be sandwiched. A malicious user could be monitoring the mempool for any autoIrrigate() or irrigate() call and:

1. Front-run, the autoIrrigate() or irrigate() call to buy beans from the Curve pool. This pushes the price of Bean in the pool up.
2. autoIrrigate() or irrigate() call is executed and Beans are bought at a higher price. This raises, once again, the Bean price.
3. The malicious user back-runs the autoIrrigate() or irrigate() call and sells the Beans at a higher price for a profit.

BVSS:

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:M/R:N/S:U (5.0)

Recommendation:

It is recommended to add any type of slippage protection during the `_swapEthForWater()` call. A possible solution would be adding a `minAmountOut` parameter to the `autoIrrigate()` and `irrigate()` functions. This function should be calculated off-chain before any `autoIrrigate()` or `irrigate()` function call.

Remediation Plan:

SOLVED: The Irrigation Protocol team solved the issue by implementing the recommended solution.

Commit ID : f573cdc385d11a8ab3b8929e97678ace5221e4d7.

4.7 (HAL-07) AUTOIRRIGATECALL WILL ALWAYS REVERT WITH OVERFLOW IF ITS CALLED WITH THE FULL REWARDAMOUNT - LOW (2.5)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

In the WaterTowerUpgradeable contract, the function autoIrrigate() is used to auto-compound the user rewards:

Listing 14: WaterTowerUpgradeable.sol (Line 84)

```
79 function autoIrrigate(address user, uint256 rewardAmount) external
↳  onlySuperAdminRole {
80     if (!WaterTowerStorage.layout().users[user].isAutoIrrigate)
↳ revert NotAutoIrrigate();
81     _irrigate(user, rewardAmount);
82     /// @dev 870391 is the gasLimit for this function
83     uint256 gasFee = 870391 * tx.gasprice;
84     WaterTowerStorage.layout().users[user].pending -= gasFee;
85     emit AutoIrrigate(user, rewardAmount, gasFee);
86 }
```

As this function is called by the protocol admins, the gas fee paid by the admins is subtracted from the user pending balance. Although, the pending balance of the user will always be zero after an _irrigate() call of the full reward amount. Hence, if autoIrrigate() is called with the full reward amount the call will always revert.

BVSS:

AO:A/AC:L/AX:L/C:N/I:L/A:N/D:N/Y:N/R:N/S:U (2.5)

Recommendation:

It is recommended to update the autoIrrigate() implementation, so it is possible to auto-compound the full reward amount. A possible implementation would be storing the gasFees to pay per user in a mapping and force the users to pay for them upon withdrawal. Note that the gasFees are inially in Ether, they should be converted to WATER before they can be substracted from the UserInfo.amount.

Remediation Plan:

RISK ACCEPTED: The Irrigation Protocol team accepted this risk, as they state that the autoIrrigate() function will always be called by an admin using full reward amount - fee amount.

4.8 (HAL-08) LACK OF A DOUBLE-STEP TRANSFEROWNERSHIP PATTERN - INFORMATIONAL (0.0)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

The OwnershipFacet contract allows transferring the ownership of the IrrigationDiamond in a single step:

Listing 15: OwnershipFacet.sol (Line 10)

```
 8 function transferOwnership(address _newOwner) external override {
 9     LibDiamond.enforceIsContractOwner();
10     LibDiamond.setContractOwner(_newOwner);
11 }
```

If the nominated EOA account is not a valid account, it is entirely possible that the owner may accidentally transfer ownership to an uncontrolled account, losing the access to all functions with the onlySuperAdminRole modifier.

BVSS:

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:F/S:C (0.0)

Recommendation:

It is recommended to implement a two-step process where the owner nominates an account and the nominated account needs to call an acceptOwnership() function for the transfer of the ownership to fully succeed. This ensures the nominated EOA account is a valid and active account. A good code example could be OpenZeppelin's Ownable2Step contract:

Listing 16: Ownable2Step.sol (Lines 52-56)

```
 1 // SPDX-License-Identifier: MIT
 2 // OpenZeppelin Contracts (last updated v4.8.0) (access/
↳ Ownable2Step.sol)
 3
 4 pragma solidity ^0.8.0;
 5
 6 import "./Ownable.sol";
 7
 8 /**
 9  * @dev Contract module which provides access control mechanism,
↳ where
10  * there is an account (an owner) that can be granted exclusive
↳ access to
11  * specific functions.
12  *
13  * By default, the owner account will be the one that deploys the
↳ contract. This
14  * can later be changed with {transferOwnership} and {
↳ acceptOwnership}.
15  *
16  * This module is used through inheritance. It will make available
↳  all functions
17  * from parent (Ownable).
18  */
19 abstract contract Ownable2Step is Ownable {
20     address private _pendingOwner;
21
22     event OwnershipTransferStarted(address indexed previousOwner,
↳ address indexed newOwner);
23
24     /**
25      * @dev Returns the address of the pending owner.
26      */
27     function pendingOwner() public view virtual returns (address)
↳ {
28         return _pendingOwner;
29     }
30
31     /**
32      * @dev Starts the ownership transfer of the contract to a new
↳  account. Replaces the pending transfer if there is one.
33      * Can only be called by the current owner.
34      */
```

```
35      function transferOwnership(address newOwner) public virtual
↳ override onlyOwner {
36          _pendingOwner = newOwner;
37          emit OwnershipTransferStarted(owner(), newOwner);
38      }
39
40      /**
41       * @dev Transfers ownership of the contract to a new account
↳ (`newOwner`) and deletes any pending owner.
42       * Internal function without access restriction.
43       */
44      function _transferOwnership(address newOwner) internal virtual
↳  override {
45          delete _pendingOwner;
46          super._transferOwnership(newOwner);
47      }
48
49      /**
50       * @dev The new owner accepts the ownership transfer.
51       */
52      function acceptOwnership() external {
53          address sender = _msgSender();
54          require(pendingOwner() == sender, "Ownable2Step: caller is
↳  not the new owner");
55          _transferOwnership(sender);
56      }
57  }
```

Remediation Plan:

ACKNOWLEDGED: The Irrigation Protocol team acknowledged this finding.

4.9 (HAL-09) CALLING SETMIDDLEASSET WOULD CAUSE THE IRRIGATION BONUS TO ALWAYS BE ZERO - INFORMATIONAL (0.0)

Commit IDs affected:
- 159927ff1e8f8e212cd09894c29c1019e5d417f6

Description:

The contract WaterTowerUpgradeable contains the function setMiddleAsset():

Listing 17: WaterTowerUpgradeable.sol (Line 270)

```
269 function setMiddleAsset(address middleAsset) external
 ↳ onlySuperAdminRole {
270     WaterTowerStorage.layout().middleAssetForIrrigate =
 ↳ middleAsset;
271 }
```

If this function is ever called to set as middleAssetForIrrigate an asset different from the BEAN token, the irrigation bonus would always be zero as currently the _swapEthForWater() would only work if WaterTowerStorage.layout().middleAssetForIrrigate == Constants.BEAN:

Listing 18: WaterTowerUpgradeable.sol (Line 196)

```
195 function _swapEthForWater(uint256 amount) internal returns (
 ↳ uint256 waterAmount) {
196     if (WaterTowerStorage.layout().middleAssetForIrrigate ==
 ↳ Constants.BEAN) {
197         /// @dev swap ETH for BEAN using curve router
198         address[9] memory route = [
199             Constants.ETHER,
200             Constants.TRI_CRYPTO_POOL,
201             Constants.USDT,
202             Constants.CURVE_BEAN_METAPOOL,
203             Constants.BEAN,
```

```
204             Constants.ZERO,
205             Constants.ZERO,
206             Constants.ZERO,
207             Constants.ZERO
208         ];
209         uint256[3][4] memory swapParams = [
210             [uint(2), 0, 3],
211             [uint(3), 0, 2],
212             [uint(0), 0, 0],
213             [uint(0), 0, 0]
214         ];
215         uint256 beanAmount = ICurveSwapRouter(Constants.
 ↳ CURVE_ROUTER).exchange_multiple{
216             value: amount
217         }(route, swapParams, amount, 0);
218 
219         waterAmount = ISprinklerUpgradeable(address(this)).
 ↳ getWaterAmount(
220             Constants.BEAN,
221             beanAmount
222         );
223     }
224 }
```

BVSS:

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:F/S:C (0.0)

Recommendation:

Consider removing the `setMiddleAsset()` function, as using it would nevertheless require a facet upgrade to work with a new `middleAssetForIrrigate`.

Remediation Plan:

ACKNOWLEDGED: The Irrigation Protocol team acknowledged this finding.

RECOMMENDATIONS OVERVIEW

1. Use the OpenZeppelin's SafeERC20 library to perform all the token transfers instead of the TransferHelper library.
2. Consider adding a try/catch code block for the safeTransfer() call in the AuctionUpgradeable._settleBid() function.
3. Set a minimum minBidAmount threshold of for example 1% of the total sellAmount in the AuctionUpgradeable contract.
4. Update the getChainlinkPrice() function as suggested, so it prevents returning any stale price.
5. Consider redesigning the logic of the AuctionUpgradeable._settleAuction() function, so such large number of iterations are not required.
6. Add any type of slippage protection during the _swapEthForWater() call. A possible solution would be adding a minAmountOut parameter to the autoIrrigate() and irrigate() functions. This function should be calculated off-chain before any autoIrrigate() or irrigate() function call.
7. Update the autoIrrigate() implementation, so it is possible to auto-compound the full reward amount. A possible implementation would be storing the gasFees to pay per user in a mapping and force the users to pay for them upon withdrawal. Note that the gasFees are inially in Ether, they should be converted to WATER before they can be substracted from the UserInfo.amount.
8. Implement a two-step process in the OwnershipFacet where the owner nominates an account and the nominated account needs to call an acceptOwnership() function for the transfer of the ownership to fully succeed.
9. Consider removing the setMiddleAsset() function from the WaterTowerUpgradeable as using it would nevertheless require a Facet upgrade to work with a new middleAssetForIrrigate.

AUTOMATED TESTING

6.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the smart contracts in scope. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified the smart contracts in the repository and was able to compile them correctly into their ABIs and binary format, Slither was run against the contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

SprinklerUpgradeable.sol

```
INFO:Detectors:
Function SprinklerUpgradeable.setTokenMultiplier(address,uint256) (contracts/core/SprinklerUpgradeable.sol#33-35) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Function TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10) contains a low level call to a custom address
Function TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16) contains a low level call to a custom address
Function TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22) contains a low level call to a custom address
Function TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
IBeanstalkUpgradeable.permitToken(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#303) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.tokenPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#321) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitERC20(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#352) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.allowancePods(address,address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#477) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.claimWithdrawal(address,uint32,uint8).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#721) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.depositPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#741) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.getDeposit(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#754) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.getWithdrawal(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#764) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.permitDeposit(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#778) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitDeposits(address,address,address[],uint256[],uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#789) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#815) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).bdv (contracts/beanstalk/IBeanstalkUpgradeable.sol#817) shadows:
        - IBeanstalkUpgradeable.bdv(address,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#602) (function)
IBeanstalkUpgradeable.withdrawDeposit(address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#829) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.plentyPerRoot(uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#864) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in SprinklerUpgradeable.depositWater(uint256) (contracts/core/SprinklerUpgradeable.sol#108-115):
        External calls:
        - TransferHelper.safeTransferFrom(address(this),msg.sender,address(this),amount) (contracts/core/SprinklerUpgradeable.sol#113)
        Event emitted after the call(s):
        - DepositWater(amount) (contracts/core/SprinklerUpgradeable.sol#114)
Reentrancy in SprinklerUpgradeable.exchangeETHToWater() (contracts/core/SprinklerUpgradeable.sol#98-106):
        External calls:
        - transferWater(waterAmount) (contracts/core/SprinklerUpgradeable.sol#103)
                - TransferHelper.safeTransfer(address(this),msg.sender,amount) (contracts/core/SprinklerUpgradeable.sol#119)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (contracts/libraries/TransferHelper.sol#14)
        Event emitted after the call(s):
        - WaterExchanged(msg.sender,Constants.ETHER,msg.value,waterAmount,false) (contracts/core/SprinklerUpgradeable.sol#105)
Reentrancy in SprinklerUpgradeable.exchangeTokenToWater(address,uint256) (contracts/core/SprinklerUpgradeable.sol#77-92):
        External calls:
        - TransferHelper.safeTransferFrom(token,msg.sender,address(this),amount) (contracts/core/SprinklerUpgradeable.sol#88)
        - transferWater(waterAmount) (contracts/core/SprinklerUpgradeable.sol#89)
                - TransferHelper.safeTransfer(address(this),msg.sender,amount) (contracts/core/SprinklerUpgradeable.sol#119)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (contracts/libraries/TransferHelper.sol#14)
        Event emitted after the call(s):
        - WaterExchanged(msg.sender,token,amount,waterAmount,false) (contracts/core/SprinklerUpgradeable.sol#91)
Reentrancy in SprinklerUpgradeable.withdrawToken(address,address,uint256) (contracts/core/SprinklerUpgradeable.sol#131-142):
        External calls:
        - TransferHelper.safeTransferETH(to,amount) (contracts/core/SprinklerUpgradeable.sol#135)
        - TransferHelper.safeTransfer(token,to,amount) (contracts/core/SprinklerUpgradeable.sol#137)
        Event emitted after the call(s):
        - WithdrawToken(token,msg.sender,amount) (contracts/core/SprinklerUpgradeable.sol#141)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Function SprinklerUpgradeable.addAssetToWhiteList(address,uint256) (contracts/core/SprinklerUpgradeable.sol#42-60) has a dubious typecast: IERC20MetadataUpgradeable<=address
Function SprinklerUpgradeable.depositWater(uint256) (contracts/core/SprinklerUpgradeable.sol#108-115) has a dubious typecast: address<=address
Function SprinklerUpgradeable.getWaterAmount(address,uint256) (contracts/core/SprinklerUpgradeable.sol#151-159) has a dubious typecast: IPriceOracleUpgradeable<=address
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
```

AUTOMATED TESTING

```
INFO:Detectors:
Setter function SprinklerUpgradeable.setTokenMultiplier(address,uint256) (contracts/core/SprinklerUpgradeable.sol#33-35) does not emit an event
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/event_setter.md
INFO:Detectors:
SprinklerStorage.layout() (contracts/core/SprinklerStorage.sol#28-33) uses assembly
        - INLINE ASM (contracts/core/SprinklerStorage.sol#30-32)
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) uses assembly
        - INLINE ASM (contracts/core/WaterCommonStorage.sol#20-22)
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10) is never used and should be removed
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/beanstalk/IBeanstalkUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/SprinklerStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/core/SprinklerUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/WaterCommonStorage.sol#3) allows old versions
Pragma version0.8.17 (contracts/interfaces/ICustomOracle.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IPriceOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/ISprinklerUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/TransferHelper.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Low level call in TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10):
        - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (contracts/libraries/TransferHelper.sol#8)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16):
        - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (contracts/libraries/TransferHelper.sol#14)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22):
        - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value)) (contracts/libraries/TransferHelper.sol#20)
Low level call in TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/libraries/TransferHelper.sol#25)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IBeanstalkUpgradeable._getPenalizedUnderlying(address,uint256,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#74-78) is not in mixedCase
Parameter SprinklerUpgradeable.setTokenMultiplier(address,uint256)._token (contracts/core/SprinklerUpgradeable.sol#33) is not in mixedCase
Parameter SprinklerUpgradeable.setTokenMultiplier(address,uint256)._multiplier (contracts/core/SprinklerUpgradeable.sol#33) is not in mixedCase
Parameter SprinklerUpgradeable.addAssetToWhiteList(address,uint256)._token (contracts/core/SprinklerUpgradeable.sol#42) is not in mixedCase
Parameter SprinklerUpgradeable.addAssetToWhiteList(address,uint256)._multiplier (contracts/core/SprinklerUpgradeable.sol#42) is not in mixedCase
Parameter SprinklerUpgradeable.unListAsset(address)._token (contracts/core/SprinklerUpgradeable.sol#66) is not in mixedCase
Parameter SprinklerUpgradeable.getWaterAmount(address,uint256)._token (contracts/core/SprinklerUpgradeable.sol#152) is not in mixedCase
Parameter SprinklerUpgradeable.getWaterAmount(address,uint256)._amount (contracts/core/SprinklerUpgradeable.sol#153) is not in mixedCase
Parameter SprinklerUpgradeable.tokenMultiplier(address)._token (contracts/core/SprinklerUpgradeable.sol#167) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function SprinklerUpgradeable.addAssetToWhiteList(address,uint256) (contracts/core/SprinklerUpgradeable.sol#42-60) contains magic number: 10
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

WaterTowerUpgradeable.sol

```
INFO:Detectors:
Function WaterTowerUpgradeable.setAutoIrrigate(bool) (contracts/core/WaterTowerUpgradeable.sol#88-91) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function WaterTowerUpgradeable.setMiddleAsset(address) (contracts/core/WaterTowerUpgradeable.sol#254-256) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function WaterTowerUpgradeable.setIrrigateBonusRate(uint256) (contracts/core/WaterTowerUpgradeable.sol#258-260) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function WaterTowerUpgradeable.setPool(uint256,uint256) (contracts/core/WaterTowerUpgradeable.sol#262-271) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
In a function WaterTowerUpgradeable.autoIrrigate(address,uint256) (contracts/core/WaterTowerUpgradeable.sol#79-86) tx.gasprice (which is set by user) variable is used. Make sure that it can not exploit the contract logic.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/tx_gasprice_warning.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Function TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10) contains a low level call to a custom address
Function TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16) contains a low level call to a custom address
Function TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22) contains a low level call to a custom address
Function TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Reentrancy in WaterTowerUpgradeable._irrigate(address,uint256) (contracts/core/WaterTowerUpgradeable.sol#122-137):
        External calls:
        - swappedWaterAmount = _swapEthForWater(rewardAmount) (contracts/core/WaterTowerUpgradeable.sol#124)
                - beanAmount = ICurveSwapRouter(Constants.CURVE_ROUTER).exchange_multiple{value: amount}(route,swapParams,amount,0) (contracts/core/WaterTowerUpgradeable.sol#200-202)
        Event emitted after the call(s):
        - Deposited(user,amount) (contracts/core/WaterTowerUpgradeable.sol#163)
                - _deposit(irrigator,totalDepositWaterAmount) (contracts/core/WaterTowerUpgradeable.sol#128)
        - Irrigate(irrigator,WaterTowerStorage.layout().middleAssetForIrrigate,rewardAmount,totalDepositWaterAmount,bonusAmount) (contracts/core/WaterTowerUpgradeable.sol#130-136)
Reentrancy in WaterTowerUpgradeable.autoIrrigate(address,uint256) (contracts/core/WaterTowerUpgradeable.sol#79-86):
        External calls:
        - _irrigate(user,rewardAmount) (contracts/core/WaterTowerUpgradeable.sol#81)
                - beanAmount = ICurveSwapRouter(Constants.CURVE_ROUTER).exchange_multiple{value: amount}(route,swapParams,amount,0) (contracts/core/WaterTowerUpgradeable.sol#200-202)
        Event emitted after the call(s):
        - AutoIrrigate(user,rewardAmount,gasFee) (contracts/core/WaterTowerUpgradeable.sol#85)
Reentrancy in WaterTowerUpgradeable.deposit(uint256,bool) (contracts/core/WaterTowerUpgradeable.sol#52-56):
        External calls:
        - IERC20Upgradeable(address(this)).safeTransferFrom(msg.sender,address(this),amount) (contracts/core/WaterTowerUpgradeable.sol#53)
```

```
        Event emitted after the call(s):
        - Deposited(user,amount) (contracts/core/WaterTowerUpgradeable.sol#163)
            - _deposit(msg.sender,amount) (contracts/core/WaterTowerUpgradeable.sol#55)
        - SetAutoIrrigate(msg.sender,block.timestamp,bAutoIrrigate) (contracts/core/WaterTowerUpgradeable.sol#90)
            - setAutoIrrigate(bAutoIrrigate) (contracts/core/WaterTowerUpgradeable.sol#54)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
WaterTowerUpgradeable.setPool(uint256,uint256) (contracts/core/WaterTowerUpgradeable.sol#262-271) uses timestamp for comparisons
        Dangerous comparisons:
        - (endTime != 0 && endTime < block.timestamp) (contracts/core/WaterTowerUpgradeable.sol#263)
        - endTime == 0 (contracts/core/WaterTowerUpgradeable.sol#265)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Function WaterTowerUpgradeable.deposit(uint256,bool) (contracts/core/WaterTowerUpgradeable.sol#52-56) has a dubious typecast: IERC20Upgradeable<address
Function WaterTowerUpgradeable.withdraw(uint256) (contracts/core/WaterTowerUpgradeable.sol#59-62) has a dubious typecast: IERC20Upgradeable<address
Function WaterTowerUpgradeable._swapEthForWater(uint256) (contracts/core/WaterTowerUpgradeable.sol#180-209) has a dubious typecast: uint256<=uint256
Function WaterTowerUpgradeable._swapEthForWater(uint256) (contracts/core/WaterTowerUpgradeable.sol#180-209) has a dubious typecast: ISprinklerUpgradeable<=address
Function WaterTowerUpgradeable.getBonusForIrrigate(uint256) (contracts/core/WaterTowerUpgradeable.sol#211-236) has a dubious typecast: uint256<=ICurveMetaPool
Function WaterTowerUpgradeable.getBonusForIrrigate(uint256) (contracts/core/WaterTowerUpgradeable.sol#211-236) has a dubious typecast: ISprinklerUpgradeable<=address
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
Setter function WaterTowerUpgradeable.setMiddleAsset(address) (contracts/core/WaterTowerUpgradeable.sol#254-256) does not emit an event
Setter function WaterTowerUpgradeable.setIrrigateBonusRate(uint256) (contracts/core/WaterTowerUpgradeable.sol#258-260) does not emit an event
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/event_setter.md
INFO:Detectors:
WaterTowerStorage.layout() (contracts/core/WaterTowerStorage.sol#51-56) uses assembly
        - INLINE ASM (contracts/core/WaterTowerStorage.sol#53-55)
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10) is never used and should be removed
TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16) is never used and should be removed
TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27) is never used and should be removed
TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/core/WaterTowerStorage.sol#3) allows old versions
Pragma version0.8.17 (contracts/core/WaterTowerUpgradeable.sol#2) allows old versions
Pragma version0.8.17 (contracts/curve/ICurveMetaPool.sol#2) allows old versions
Pragma version0.8.17 (contracts/curve/ICurveSwapRouter.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IPriceOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/ISprinklerUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IWaterTowerUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/TransferHelper.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in WaterTowerUpgradeable.claim(uint256) (contracts/core/WaterTowerUpgradeable.sol#65-69):
        - (success) = msg.sender.call{value: claimAmount}() (contracts/core/WaterTowerUpgradeable.sol#67)
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Low level call in TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10):
        - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (contracts/libraries/TransferHelper.sol#8)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16):
        - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (contracts/libraries/TransferHelper.sol#14)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22):
        - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value)) (contracts/libraries/TransferHelper.sol#20)
Low level call in TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/libraries/TransferHelper.sol#25)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function ICurveMetaPool.get_price_cumulative_last() (contracts/curve/ICurveMetaPool.sol#5) is not in mixedCase
Function ICurveMetaPool.get_dy(int128,int128,uint256,uint256[2]) (contracts/curve/ICurveMetaPool.sol#7-12) is not in mixedCase
Function ICurveMetaPool.get_dy(int128,int128,uint256) (contracts/curve/ICurveMetaPool.sol#14) is not in mixedCase
Function ICurveMetaPool.get_dy(uint256,uint256,uint256) (contracts/curve/ICurveMetaPool.sol#15) is not in mixedCase
Function ICurveMetaPool.get_dy_underlying(int128,int128,uint256) (contracts/curve/ICurveMetaPool.sol#17) is not in mixedCase
Function ICurveMetaPool.get_dy_underlying(int128,int128,uint256,uint256[2]) (contracts/curve/ICurveMetaPool.sol#19-24) is not in mixedCase
Function ICurveMetaPool.get_virtual_price() (contracts/curve/ICurveMetaPool.sol#26) is not in mixedCase
Function ICurveSwapRouter.exchange_multiple(address[9],uint256[3][4],uint256,uint256) (contracts/curve/ICurveSwapRouter.sol#5-10) is not in mixedCase
Parameter ICurveSwapRouter.exchange_multiple(address[9],uint256[3][4],uint256,uint256)._swap_params (contracts/curve/ICurveSwapRouter.sol#7) is not in mixedCase
Function ICurveSwapRouter.get_exchange_multiple_amount(address[9],uint256[3][4],uint256) (contracts/curve/ICurveSwapRouter.sol#12-16) is not in mixedCase
Parameter ICurveSwapRouter.get_exchange_multiple_amount(address[9],uint256[3][4],uint256)._swap_params (contracts/curve/ICurveSwapRouter.sol#14) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function WaterTowerUpgradeable.autoIrrigate(address,uint256) (contracts/core/WaterTowerUpgradeable.sol#79-86) contains magic number: 870391
Function WaterTowerUpgradeable.getBonusForIrrigate(uint256) (contracts/core/WaterTowerUpgradeable.sol#211-236) contains magic number: 3
Function WaterTowerUpgradeable.setPool(uint256,uint256) (contracts/core/WaterTowerUpgradeable.sol#262-271) contains magic number: 2592000
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

TrancheBondUpgradeable.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function TrancheBondUpgradeable.onlyWaterHolder() (contracts/core/TrancheBondUpgradeable.sol#367-371) is an unprotected initializer.
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
TrancheBondUpgradeable.receivePodsForTranche(uint256).startOffset (contracts/core/TrancheBondUpgradeable.sol#168) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
```

AUTOMATED TESTING

```
TrancheBondUpgradeable.receivePodsForTranche(uint256).startIndex (contracts/core/TrancheBondUpgradeable.sol#167) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
TrancheBondUpgradeable._plotsForTranche(uint256,uint256,uint256).i (contracts/core/TrancheBondUpgradeable.sol#336) is a local variable never initialized
TrancheBondUpgradeable.createTranchesWithPods(uint256[],uint256[],uint256[],uint256).i (contracts/core/TrancheBondUpgradeable.sol#88) is a local variable never initialized
TrancheBondUpgradeable.receivePodsForTranche(uint256).i (contracts/core/TrancheBondUpgradeable.sol#172) is a local variable never initialized
TrancheBondUpgradeable.receivePodsForTranche(uint256).totalPods (contracts/core/TrancheBondUpgradeable.sol#170) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
IBeanstalkUpgradeable.permitToken(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#303) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.tokenPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#321) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitERC20(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#352) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.allowancePods(address,address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#477) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.claimWithdrawal(address,uint32,uint8).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#721) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.depositPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#741) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.getDeposit(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#754) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.getWithdrawal(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#764) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.permitDeposit(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#778) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitDeposits(address,address,address[],uint256[],uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#789) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#815) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).bdv (contracts/beanstalk/IBeanstalkUpgradeable.sol#817) shadows:
        - IBeanstalkUpgradeable.bdv(address,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#602) (function)
IBeanstalkUpgradeable.withdrawDeposit(address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#829) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.plentyPerRoot(uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#864) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
TrancheBondUpgradeable.createTranchesWithPods(uint256[],uint256[],uint256[],uint256) (contracts/core/TrancheBondUpgradeable.sol#72-145) has external calls inside a loop: WaterCommonStorage.layout().beanstalk.transferPlot(msg.sender,addre
ss(this),_index,_start,_end) (contracts/core/TrancheBondUpgradeable.sol#95-101)
TrancheBondUpgradeable.createTranchesWithPods(uint256[],uint256[],uint256[],uint256) (contracts/core/TrancheBondUpgradeable.sol#72-145) has external calls inside a loop: _fmvInPlot = (IPodsOracleUpgradeable(address(this)).latestPriceOfPo
ds(id,amount) * beanPrice) / 1e18 (contracts/core/TrancheBondUpgradeable.sol#113-116)
TrancheBondUpgradeable.receivePodsForTranche(uint256) (contracts/core/TrancheBondUpgradeable.sol#148-236) has external calls inside a loop: WaterCommonStorage.layout().beanstalk.transferPlot(address(this),msg.sender,originalPodIndex + of
fset,start,amount + start) (contracts/core/TrancheBondUpgradeable.sol#198-204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop
INFO:Detectors:
Reentrancy in TrancheBondUpgradeable.createTranchesWithPods(uint256[],uint256[],uint256[],uint256) (contracts/core/TrancheBondUpgradeable.sol#72-145):
        External calls:
        - WaterCommonStorage.layout().beanstalk.transferPlot(msg.sender,address(this),_index,_start,_end) (contracts/core/TrancheBondUpgradeable.sol#95-101)
        - _fmvInPlot = (IPodsOracleUpgradeable(address(this)).latestPriceOfPods(id,amount) * beanPrice) / 1e18 (contracts/core/TrancheBondUpgradeable.sol#113-116)
        - _createTranchesFromDeposit(curDepositCount,totalFMV,msg.sender) (contracts/core/TrancheBondUpgradeable.sol#142)
                - IERC1155WhitelistUpgradeable(address(this)).mint(owner,trancheId,fmv,) (contracts/core/TrancheBondUpgradeable.sol#254)
        Event emitted after the call(s):
        - CreateTranche(curDepositCount,msg.sender,totalFMV,block.timestamp,beanPrice) (contracts/core/TrancheBondUpgradeable.sol#144)
Reentrancy in TrancheBondUpgradeable.receivePodsForTranche(uint256) (contracts/core/TrancheBondUpgradeable.sol#148-236):
        External calls:
        - WaterCommonStorage.layout().beanstalk.transferPlot(address(this),msg.sender,originalPodIndex + offset,start,amount + start) (contracts/core/TrancheBondUpgradeable.sol#198-204)
        - IERC1155BurnableUpgradeable(address(this)).burn(msg.sender,_trancheId,fmv) (contracts/core/TrancheBondUpgradeable.sol#233)
        Event emitted after the call(s):
        - ReceivePodsWithTranche(_trancheId,msg.sender,receivePodIndexes,totalPods) (contracts/core/TrancheBondUpgradeable.sol#234)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
TrancheBondUpgradeable.receivePodsForTranche(uint256) (contracts/core/TrancheBondUpgradeable.sol#148-236) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < underlyingAsset.maturityDate (contracts/core/TrancheBondUpgradeable.sol#155)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Function TrancheBondUpgradeable.createTranchesWithPods(uint256[],uint256[],uint256[],uint256) (contracts/core/TrancheBondUpgradeable.sol#72-145) has a dubious typecast: IPodsOracleUpgradeable<=address
Function TrancheBondUpgradeable.receivePodsForTranche(uint256) (contracts/core/TrancheBondUpgradeable.sol#148-236) has a dubious typecast: IERC1155Upgradeable<=address
Function TrancheBondUpgradeable.receivePodsForTranche(uint256) (contracts/core/TrancheBondUpgradeable.sol#148-236) has a dubious typecast: IERC1155BurnableUpgradeable<=address
Function TrancheBondUpgradeable._createTranchesFromDeposit(uint256,uint256,address) (contracts/core/TrancheBondUpgradeable.sol#238-259) has a dubious typecast: IERC1155WhitelistUpgradeable<=address
Function TrancheBondUpgradeable.getPlotsForUser(uint256,address) (contracts/core/TrancheBondUpgradeable.sol#284-291) has a dubious typecast: IERC1155Upgradeable<=address
Function FullMath.mulDivRoundingUp128(uint128,uint128,uint256) (contracts/libraries/FullMath.sol#23-33) has a dubious typecast: uint256<=uint256
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
TrancheBondStorage.layout() (contracts/core/TrancheBondStorage.sol#66-71) uses assembly
        - INLINE ASM (contracts/core/TrancheBondStorage.sol#68-70)
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) uses assembly
        - INLINE ASM (contracts/core/WaterCommonStorage.sol#20-22)
WaterTowerStorage.layout() (contracts/core/WaterTowerStorage.sol#51-56) uses assembly
        - INLINE ASM (contracts/core/WaterTowerStorage.sol#53-55)
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
FullMath.mulDivRoundingUp(uint256,uint256,uint256) (contracts/libraries/FullMath.sol#11-21) is never used and should be removed
FullMath.mulDivRoundingUp128(uint128,uint128,uint256) (contracts/libraries/FullMath.sol#23-33) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
WaterTowerStorage.curPool() (contracts/core/WaterTowerStorage.sol#58-60) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/beanstalk/IBeanstalkUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/TrancheBondStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/core/TrancheBondUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/WaterCommonStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/core/WaterTowerStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IBeanstalkPrice.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IERC1155WhitelistUpgradeable.sol#3) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IPodsOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/FullMath.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/PodTransferHelper.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IBeanstalkUpgradeable._getPenalizedUnderlying(address,uint256,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#74-78) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function TrancheBondUpgradeable.receivePodsForTranche(uint256) (contracts/core/TrancheBondUpgradeable.sol#148-236) contains magic number: 3
Function TrancheBondUpgradeable._createTranchesFromDeposit(uint256,uint256,address) (contracts/core/TrancheBondUpgradeable.sol#238-259) contains magic numbers: 20, 3
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Function PodTransferHelper.getPlotWithOffset(uint256,uint256,uint256,uint256,uint256,uint128[6]) (contracts/libraries/PodTransferHelper.sol#39-72) contains magic number: 3
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

AuctionUpgradeable.sol

```
INFO:Detectors:
Function AuctionUpgradeable.setPurchaseToken(address,bool) (contracts/core/AuctionUpgradeable.sol#410-412) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function AuctionUpgradeable.setSellToken(address,bool) (contracts/core/AuctionUpgradeable.sol#415-417) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function AuctionUpgradeable.setAuctionFee(uint256,address) (contracts/core/AuctionUpgradeable.sol#419-427) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) performs a multiplication on the result of a division:
        - feeAmount = (((sellAmount * 1e30) / ethPrice) * AuctionStorage.layout().feeNumerator) / FEE_DENOMINATOR (contracts/core/AuctionUpgradeable.sol#133-134)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Function TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10) contains a low level call to a custom address
Function TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16) contains a low level call to a custom address
Function TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22) contains a low level call to a custom address
Function TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Reentrancy in AuctionUpgradeable._settleAuction(uint256) (contracts/core/AuctionUpgradeable.sol#308-368):
        External calls:
        - TransferHelper.safeTransfer(auction.sellToken,auction.seller,auction.reserve) (contracts/core/AuctionUpgradeable.sol#314)
        - IERC1155Upgradeable(address(this)).safeTransferFrom(address(this),auction.seller,trancheIndex,auction.reserve,Constants.EMPTY) (contracts/core/AuctionUpgradeable.sol#316-322)
        Event emitted after the call(s):
        - AuctionClosed(auction.reserve,auctionId) (contracts/core/AuctionUpgradeable.sol#327)
Reentrancy in AuctionUpgradeable._settleAuction(uint256) (contracts/core/AuctionUpgradeable.sol#308-368):
        External calls:
        - settledAmount = _settleBid(auction.sellToken,auction.trancheIndex,auction.seller,bid,availableAmount) (contracts/core/AuctionUpgradeable.sol#335-341)
                - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (contracts/libraries/TransferHelper.sol#14)
                - TransferHelper.safeTransfer(bid.purchaseToken,bid.bidder,getPayAmount(bid.purchaseToken,repayAmount,bid.bidPrice,sellToken)) (contracts/core/AuctionUpgradeable.sol#383-387)
                - TransferHelper.safeTransfer(sellToken,bid.bidder,settledAmount) (contracts/core/AuctionUpgradeable.sol#390)
                - IERC1155Upgradeable(address(this)).safeTransferFrom(address(this),bid.bidder,trancheIndex,settledAmount,Constants.EMPTY) (contracts/core/AuctionUpgradeable.sol#392-398)
                - TransferHelper.safeTransfer(bid.purchaseToken,seller,getPayAmount(bid.purchaseToken,settledAmount,bid.bidPrice,sellToken)) (contracts/core/AuctionUpgradeable.sol#401-405)
        - TransferHelper.safeTransfer(auction.sellToken,auction.seller,availableAmount) (contracts/core/AuctionUpgradeable.sol#353)
        - IERC1155Upgradeable(address(this)).safeTransferFrom(address(this),auction.seller,trancheIndex,availableAmount,Constants.EMPTY) (contracts/core/AuctionUpgradeable.sol#355-361)
        Event emitted after the call(s):
        - AuctionClosed(availableAmount,auctionId) (contracts/core/AuctionUpgradeable.sol#367)
Reentrancy in AuctionUpgradeable.buyNow(uint256,uint128,address) (contracts/core/AuctionUpgradeable.sol#191-237):
        External calls:
        - TransferHelper.safeTransferFrom(purchaseToken,msg.sender,auction.seller,payAmount) (contracts/core/AuctionUpgradeable.sol#214)
        - TransferHelper.safeTransfer(auction.sellToken,msg.sender,purchaseAmount) (contracts/core/AuctionUpgradeable.sol#217)
        - IERC1155Upgradeable(address(this)).safeTransferFrom(address(this),msg.sender,trancheIndex,purchaseAmount,Constants.EMPTY) (contracts/core/AuctionUpgradeable.sol#219-225)
        Event emitted after the call(s):
        - AuctionBuy(msg.sender,payAmount,purchaseAmount,auction.sellToken,auction.trancheIndex,purchaseToken,auctionId) (contracts/core/AuctionUpgradeable.sol#228-236)
Reentrancy in AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189):
        External calls:
        - TransferHelper.safeTransferFrom(sellToken,msg.sender,address(this),(uint256(sellAmount) * (FEE_DENOMINATOR + AuctionStorage.layout().feeNumerator)) / FEE_DENOMINATOR) (contracts/core/AuctionUpgradeable.sol#114-120)
        - IERC1155Upgradeable(address(this)).safeTransferFrom(msg.sender,address(this),trancheIndex,uint256(sellAmount),Constants.EMPTY) (contracts/core/AuctionUpgradeable.sol#123-129)
        - IWaterTowerUpgradeable(address(this)).addETHReward{value: feeAmount}() (contracts/core/AuctionUpgradeable.sol#137)
        External calls sending eth:
        - address(msg.sender).transfer(msg.value - feeAmount) (contracts/core/AuctionUpgradeable.sol#136)
        - IWaterTowerUpgradeable(address(this)).addETHReward{value: feeAmount}() (contracts/core/AuctionUpgradeable.sol#137)
        Event emitted after the call(s):
        - AuctionCreated(msg.sender,_startTime,_duration,_sellToken,_trancheIndex,_sellAmount,_minBidAmount,_fixedPrice,_priceRangeStart,_priceRangeEnd,_auctionType,_assetType,AuctionStorage.layout().currentAuctionId) (contracts/core/Auc
tionUpgradeable.sol#173-187)
Reentrancy in AuctionUpgradeable.placeBid(uint256,uint128,address,uint128) (contracts/core/AuctionUpgradeable.sol#239-279):
        External calls:
        - TransferHelper.safeTransferFrom(purchaseToken,msg.sender,address(this),payAmount) (contracts/core/AuctionUpgradeable.sol#266)
        Event emitted after the call(s):
        - AuctionBid(msg.sender,bidAmount,purchaseToken,bidPrice,auctionId,currentBidId) (contracts/core/AuctionUpgradeable.sol#277)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(startTime == 0 || startTime >= block.timestamp,start time must be in the future) (contracts/core/AuctionUpgradeable.sol#140)
AuctionUpgradeable.closeAuction(uint256) (contracts/core/AuctionUpgradeable.sol#281-291) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(auction.seller != address(0) && currentTime >= auction.startTime + auction.duration && auction.status != AuctionStatus.Closed,auction can't be closed) (contracts/core/AuctionUpgradeable.sol#284-289)
AuctionUpgradeable._checkAuctionRangeTime(uint256,uint256) (contracts/core/AuctionUpgradeable.sol#478-487) uses timestamp for comparisons
        Dangerous comparisons:
        - startTime > 0 && startTime > currentTime (contracts/core/AuctionUpgradeable.sol#480)
        - endTime > 0 && endTime <= currentTime (contracts/core/AuctionUpgradeable.sol#483)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Function AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) has a dubious typecast: uint256<=address
Function AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) has a dubious typecast: IERC1155Upgradeable<=address
Function AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) has a dubious typecast: IPriceOracleUpgradeable<=address
Function AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) has a dubious typecast: IWaterTowerUpgradeable<=address
Function AuctionUpgradeable.buyNow(uint256,uint128,address) (contracts/core/AuctionUpgradeable.sol#191-237) has a dubious typecast: IERC1155Upgradeable<=address
Function AuctionUpgradeable._settleAuction(uint256) (contracts/core/AuctionUpgradeable.sol#308-368) has a dubious typecast: IERC1155Upgradeable<=address
Function AuctionUpgradeable._settleBid(address,uint256,address,Bid,uint128) (contracts/core/AuctionUpgradeable.sol#371-406) has a dubious typecast: IERC1155Upgradeable<=address
Function FullMath.mulDivRoundingUp128(uint128,uint128,uint256) (contracts/libraries/FullMath.sol#23-33) has a dubious typecast: uint256<=uint256
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
Setter function AuctionUpgradeable.setPurchaseToken(address,bool) (contracts/core/AuctionUpgradeable.sol#410-412) does not emit an event
Setter function AuctionUpgradeable.setSellToken(address,bool) (contracts/core/AuctionUpgradeable.sol#415-417) does not emit an event
Setter function AuctionUpgradeable.setAuctionFee(uint256,address) (contracts/core/AuctionUpgradeable.sol#419-427) does not emit an event
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/event_setter.md
INFO:Detectors:
AuctionStorage.layout() (contracts/core/AuctionStorage.sol#63-68) uses assembly
        - INLINE ASM (contracts/core/AuctionStorage.sol#65-67)
TrancheBondStorage.layout() (contracts/core/TrancheBondStorage.sol#66-71) uses assembly
        - INLINE ASM (contracts/core/TrancheBondStorage.sol#68-70)
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
FullMath.mulDivRoundingUp(uint256,uint256,uint256) (contracts/libraries/FullMath.sol#11-21) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
TrancheBondStorage.layout() (contracts/core/TrancheBondStorage.sol#66-71) is never used and should be removed
TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10) is never used and should be removed
TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
INFO:Detectors:
Pragma version^0.8.17 (contracts/core/AuctionStorage.sol#3) allows old versions
Pragma version0.8.17 (contracts/core/AuctionUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/TrancheBondStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IPriceOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IWaterTowerUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/FullMath.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/TransferHelper.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Low level call in TransferHelper.safeApprove(address,address,uint256) (contracts/libraries/TransferHelper.sol#6-10):
        - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (contracts/libraries/TransferHelper.sol#8)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (contracts/libraries/TransferHelper.sol#12-16):
        - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (contracts/libraries/TransferHelper.sol#14)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256) (contracts/libraries/TransferHelper.sol#18-22):
        - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value)) (contracts/libraries/TransferHelper.sol#20)
Low level call in TransferHelper.safeTransferETH(address,uint256) (contracts/libraries/TransferHelper.sol#24-27):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/libraries/TransferHelper.sol#25)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter AuctionUpgradeable.setPurchaseToken(address,bool)._token (contracts/core/AuctionUpgradeable.sol#410) is not in mixedCase
Parameter AuctionUpgradeable.setPurchaseToken(address,bool)._bEnable (contracts/core/AuctionUpgradeable.sol#410) is not in mixedCase
Parameter AuctionUpgradeable.setSellToken(address,bool)._token (contracts/core/AuctionUpgradeable.sol#415) is not in mixedCase
Parameter AuctionUpgradeable.setSellToken(address,bool)._bEnable (contracts/core/AuctionUpgradeable.sol#415) is not in mixedCase
Parameter AuctionUpgradeable.setAuctionFee(uint256,address)._newFeeNumerator (contracts/core/AuctionUpgradeable.sol#420) is not in mixedCase
Parameter AuctionUpgradeable.setAuctionFee(uint256,address)._newFeeReceiver (contracts/core/AuctionUpgradeable.sol#421) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Reentrancy in AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189):
        External calls:
        - address(msg.sender).transfer(msg.value - feeAmount) (contracts/core/AuctionUpgradeable.sol#136)
        External calls sending eth:
        - address(msg.sender).transfer(msg.value - feeAmount) (contracts/core/AuctionUpgradeable.sol#136)
        - IWaterTowerUpgradeable(address(this)).addETHReward{value: feeAmount}() (contracts/core/AuctionUpgradeable.sol#137)
        Event emitted after the call(s):
        - AuctionCreated(msg.sender,_startTime,_duration,_sellToken,_trancheIndex,_sellAmount,_minBidAmount,_fixedPrice,_priceRangeStart,_priceRangeEnd,_auctionType,_assetType,AuctionStorage.layout().currentAuctionId) (contracts/core/AuctionUpgradeable.sol#173-187)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function AuctionUpgradeable.createAuction(uint96,uint96,address,uint256,uint128,uint128,uint128,uint128,AuctionType) (contracts/core/AuctionUpgradeable.sol#100-189) contains magic numbers: 3, 1e30
Function AuctionUpgradeable.setAuctionFee(uint256,address) (contracts/core/AuctionUpgradeable.sol#419-427) contains magic number: 25
Function AuctionUpgradeable.getPayAmount(address,uint128,uint128,address) (contracts/core/AuctionUpgradeable.sol#435-446) contains magic numbers: 10, 10
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

ERC1155WhitelistUpgradeable.sol

```
INFO:Detectors:
Function ERC1155WhitelistUpgradeable.setTokenBaseURI(string) (contracts/core/ERC1155WhitelistUpgradeable.sol#200-202) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function ERC1155WhitelistUpgradeable.setContractURI(string) (contracts/core/ERC1155WhitelistUpgradeable.sol#204-206) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
IBeanstalkUpgradeable.permitToken(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#303) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.tokenPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#321) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitERC20(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#352) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.allowancePods(address,address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#477) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.claimWithdrawal(address,uint32,uint8).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#721) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.depositPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#741) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.getDeposit(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#754) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.getWithdrawal(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#764) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.permitDeposit(address,address,address,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#778) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitDeposits(address,address,address[],uint256[],uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#789) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#815) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).bdv (contracts/beanstalk/IBeanstalkUpgradeable.sol#817) shadows:
        - IBeanstalkUpgradeable.bdv(address,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#602) (function)
IBeanstalkUpgradeable.withdrawDeposit(address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#829) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.plentyPerRoot(uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#864) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Function FullMath.mulDivRoundingUp128(uint128,uint128,uint256) (contracts/libraries/FullMath.sol#23-33) has a dubious typecast: uint256<=uint256
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
Setter function ERC1155WhitelistUpgradeable.setTokenBaseURI(string) (contracts/core/ERC1155WhitelistUpgradeable.sol#200-202) does not emit an event
Setter function ERC1155WhitelistUpgradeable.setContractURI(string) (contracts/core/ERC1155WhitelistUpgradeable.sol#204-206) does not emit an event
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/event_setter.md
INFO:Detectors:
beforeTokenTransfer in ERC1155WhitelistUpgradeable._beforeTokenTransfer(address,address,address,uint256[],uint256[],bytes) (contracts/core/ERC1155WhitelistUpgradeable.sol#40-65) must have virtual and super.
Reference: https://docs.openzeppelin.com/contracts/4.x/extending-contracts#rules_of_hooks
INFO:Detectors:
ERC1155WhitelistStorage.layout() (contracts/core/ERC1155WhitelistStorage.sol#21-26) uses assembly
        - INLINE ASM (contracts/core/ERC1155WhitelistStorage.sol#23-25)
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) uses assembly
        - INLINE ASM (contracts/core/WaterCommonStorage.sol#20-22)
```

```
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
ERC1155WhitelistUpgradeable.supportsInterface(bytes4) (contracts/core/ERC1155WhitelistUpgradeable.sol#68-80) compares to a boolean constant:
        -(ERC1155Upgradeable.supportsInterface(interfaceId) || AccessControlEnumerableUpgradeable.supportsInterface(interfaceId) || (LibDiamond.diamondStorage().supportedInterfaces[interfaceId] == true)) (contracts/core/ERC1155WhitelistU
pgradeable.sol#77-79)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
FullMath.mulDivRoundingUp(uint256,uint256,uint256) (contracts/libraries/FullMath.sol#11-21) is never used and should be removed
FullMath.mulDivRoundingUp128(uint128,uint128,uint256) (contracts/libraries/FullMath.sol#23-33) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/beanstalk/IBeanstalkUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/ERC1155WhitelistStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/core/ERC1155WhitelistUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/WaterCommonStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IERC1155WhitelistUpgradeable.sol#3) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IPodsOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/FullMath.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall1(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IBeanstalkUpgradeable._getPenalizedUnderlying(address,uint256,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#74-78) is not in mixedCase
Parameter ERC1155WhitelistUpgradeable.setTokenBaseURI(string)._uri (contracts/core/ERC1155WhitelistUpgradeable.sol#200) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

WaterCommonUpgradeable.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
IBeanstalkUpgradeable.permitToken(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#303) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.tokenPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#321) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitERC20(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#352) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.allowancePods(address,address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#477) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.claimWithdrawal(address,uint32,uint8).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#721) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.depositPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#741) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.getDeposit(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#754) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.getWithdrawal(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#764) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.permitDeposit(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#778) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitDeposits(address,address,address[],uint256[],uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#789) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#815) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).bdv (contracts/beanstalk/IBeanstalkUpgradeable.sol#817) shadows:
        - IBeanstalkUpgradeable.bdv(address,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#602) (function)
IBeanstalkUpgradeable.withdrawDeposit(address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#829) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.plenyPerRoot(uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#864) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) uses assembly
        - INLINE ASM (contracts/core/WaterCommonStorage.sol#20-22)
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)

LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

```
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/beanstalk/IBeanstalkUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/WaterCommonStorage.sol#3) allows old versions
Pragma version0.8.17 (contracts/core/WaterCommonUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IBeanstalkUpgradeable._getPenalizedUnderlying(address,uint256,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#74-78) is not in mixedCase
Function WaterCommonUpgradeable.WaterCommon_Initialize(address,address) (contracts/core/WaterCommonUpgradeable.sol#15-17) is not in mixedCase
Parameter WaterCommonUpgradeable.WaterCommon_Initialize(address,address)._beanstalk (contracts/core/WaterCommonUpgradeable.sol#15) is not in mixedCase
Parameter WaterCommonUpgradeable.WaterCommon_Initialize(address,address)._fertilizer (contracts/core/WaterCommonUpgradeable.sol#15) is not in mixedCase
Function WaterCommonUpgradeable.__WaterCommon_init(address,address) (contracts/core/WaterCommonUpgradeable.sol#19-29) is not in mixedCase
Parameter WaterCommonUpgradeable.__WaterCommon_init(address,address)._beanstalk (contracts/core/WaterCommonUpgradeable.sol#19) is not in mixedCase
Parameter WaterCommonUpgradeable.__WaterCommon_init(address,address)._fertilizer (contracts/core/WaterCommonUpgradeable.sol#19) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

PriceOracleUpgradeable.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function PriceOracleUpgradeable.setDirectPrice(address,uint256) (contracts/oracles/PriceOracleUpgradeable.sol#73-76) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function PriceOracleUpgradeable.setOracle(address,address,address,OracleType) (contracts/oracles/PriceOracleUpgradeable.sol#79-99) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
PriceOracleUpgradeable.getUniswapV3Price(address,OracleItem) (contracts/oracles/PriceOracleUpgradeable.sol#59-69) performs a multiplication on the result of a division:
        - UniswapV3Twap.getTwap(Constants.D18,asset,oracleItem.oracle,10) * (oracleItem.multiplier / Constants.D18) (contracts/oracles/PriceOracleUpgradeable.sol#65-67)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Function PriceOracleUpgradeable.setOracle(address,address,address,OracleType) (contracts/oracles/PriceOracleUpgradeable.sol#79-99) has a dubious typecast: AggregatorV2V3Interface<=uint256
Function UniswapV3Twap.getSqrtTwapX96(address,uint32) (contracts/oracles/uniswapv3/UniswapV3Twap.sol#11-30) has a dubious typecast: int56<=int32
Function UniswapV3Twap.getTwap(uint256,address,address,uint32) (contracts/oracles/uniswapv3/UniswapV3Twap.sol#32-47) has a dubious typecast: uint256<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
PriceOracleStorage.layout() (contracts/oracles/PriceOracleStorage.sol#34-39) uses assembly
        - INLINE ASM (contracts/oracles/PriceOracleStorage.sol#36-38)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.17 (contracts/interfaces/ICustomOracle.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/oracles/AggregatorV2V3Interface.sol#3) allows old versions
Pragma version^0.8.17 (contracts/oracles/PriceOracleStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/oracles/PriceOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.0 (contracts/oracles/uniswapv3/UniswapV3Twap.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

```
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
UniswapV3Twap.slitherConstructorConstantVariables() (contracts/oracles/uniswapv3/UniswapV3Twap.sol#8-48) uses literals with too many digits:
        - Q192 = 0x1000000000000000000000000000000000000000000000000 (contracts/oracles/uniswapv3/UniswapV3Twap.sol#9)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Function PriceOracleUpgradeable.getUnderlyingPriceETH() (contracts/oracles/PriceOracleUpgradeable.sol#26-29) contains magic number: 1e10
Function PriceOracleUpgradeable.getUniswapV3Price(address,OracleItem) (contracts/oracles/PriceOracleUpgradeable.sol#59-69) contains magic numbers: 10, 10
Function PriceOracleUpgradeable.setOracle(address,address,address,OracleType) (contracts/oracles/PriceOracleUpgradeable.sol#79-99) contains magic numbers: 10, 10, 10
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

PodsOracleUpgradeable.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
IBeanstalkUpgradeable.permitToken(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#303) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.tokenPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#321) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitERC20(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#352) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.allowancePods(address,address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#477) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.claimWithdrawal(address,uint32,uint8).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#721) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.depositPermitNonces(address).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#741) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.getDeposit(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#754) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.getWithdrawal(address,address,uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#764) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.permitDeposit(address,address,address,uint256,uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#778) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.permitDeposits(address,address,address[],uint256[],uint256,uint8,bytes32,bytes32).owner (contracts/beanstalk/IBeanstalkUpgradeable.sol#789) shadows:
        - IBeanstalkUpgradeable.owner() (contracts/beanstalk/IBeanstalkUpgradeable.sol#150) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#815) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.transferDeposit(address,address,address,uint32,uint256).bdv (contracts/beanstalk/IBeanstalkUpgradeable.sol#817) shadows:
        - IBeanstalkUpgradeable.bdv(address,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#602) (function)
IBeanstalkUpgradeable.withdrawDeposit(address,uint32,uint256).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#829) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
IBeanstalkUpgradeable.plentyPerRoot(uint32).season (contracts/beanstalk/IBeanstalkUpgradeable.sol#864) shadows:
        - IBeanstalkUpgradeable.season() (contracts/beanstalk/IBeanstalkUpgradeable.sol#870) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
WaterCommonStorage.layout() (contracts/core/WaterCommonStorage.sol#18-23) uses assembly
        - INLINE ASM (contracts/core/WaterCommonStorage.sol#20-22)
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/beanstalk/IBeanstalkUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/core/WaterCommonStorage.sol#3) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version0.8.17 (contracts/interfaces/IPodsOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version0.8.17 (contracts/libraries/Oracle/LibPrice.sol#2) allows old versions
Pragma version^0.8.17 (contracts/oracles/PodsOracleUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IBeanstalkUpgradeable._getPenalizedUnderlying(address,uint256,uint256) (contracts/beanstalk/IBeanstalkUpgradeable.sol#74-78) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
PodsOracleUpgradeable.ONE (contracts/oracles/PodsOracleUpgradeable.sol#12) is never used in PodsOracleUpgradeable (contracts/oracles/PodsOracleUpgradeable.sol#10-34)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
In a function LibPrice.getPriceOfPods(uint256,uint256,uint256,uint256) (contracts/libraries/Oracle/LibPrice.sol#19-36) variable LibPrice.BDV_FACTOR (contracts/libraries/Oracle/LibPrice.sol#7) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

WaterUpgradeable.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version0.8.17 (contracts/tokens/WaterUpgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Function WaterUpgradeable.Water_Initialize() (contracts/tokens/WaterUpgradeable.sol#10-13) is not in mixedCase
Function WaterUpgradeable.__Water_init() (contracts/tokens/WaterUpgradeable.sol#18-21) is not in mixedCase
Function WaterUpgradeable.__Water_init_unchained() (contracts/tokens/WaterUpgradeable.sol#23-25) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
WaterUpgradeable.__Water_init_unchained() (contracts/tokens/WaterUpgradeable.sol#23-25) uses literals with too many digits:
        - _mint(msg.sender,100000000e18) (contracts/tokens/WaterUpgradeable.sol#24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Function WaterUpgradeable.__Water_init_unchained() (contracts/tokens/WaterUpgradeable.sol#23-25) contains magic number: 100000000e18
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

IrrigationDiamond.sol

```
INFO:Detectors:
Function Diamond.constructor(address,address) (contracts/utils/Diamond.sol#15-28) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function IrrigationDiamond.constructor(address,address) (contracts/IrrigationDiamond.sol#15-26) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
Token contract has a Diamond.fallback() (contracts/utils/Diamond.sol#32-59) which might be dangerous to implement
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/token_fallback.md
```

AUTOMATED TESTING

```
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function Diamond.fallback() (contracts/utils/Diamond.sol#32-59) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Diamond.fallback() (contracts/utils/Diamond.sol#32-59) uses assembly
        - INLINE ASM (contracts/utils/Diamond.sol#36-38)
        - INLINE ASM (contracts/utils/Diamond.sol#43-58)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/IrrigationDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondLoupe.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/Diamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/ERC2535Initializable.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IrrigationDiamond.FinalizeInitialization() (contracts/IrrigationDiamond.sol#35-38) is not in mixedCase
Function IrrigationDiamond.FinalReInitialize(uint8) (contracts/IrrigationDiamond.sol#40-43) is not in mixedCase
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier ERC2535Initializable.ERC2535Initializer() (contracts/utils/ERC2535Initializable.sol#11-19) is not in mixedCase
Modifier ERC2535Initializable.ERC2535Reinitializer(uint8) (contracts/utils/ERC2535Initializable.sol#21-29) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

Diamond.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Function Diamond.constructor(address,address) (contracts/utils/Diamond.sol#35-28) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Function Diamond.fallback() (contracts/utils/Diamond.sol#32-59) has a dubious typecast: address<=bytes20
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Diamond.fallback() (contracts/utils/Diamond.sol#32-59) uses assembly
        - INLINE ASM (contracts/utils/Diamond.sol#36-38)
        - INLINE ASM (contracts/utils/Diamond.sol#43-58)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/Diamond.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
```

```
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

DiamondCutFacet.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
DiamondCutFacet.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/utils/DiamondCutFacet.sol#40) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/DiamondCutFacet.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter DiamondCutFacet.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/utils/DiamondCutFacet.sol#23) is not in mixedCase
Parameter DiamondCutFacet.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/utils/DiamondCutFacet.sol#24) is not in mixedCase
Parameter DiamondCutFacet.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/utils/DiamondCutFacet.sol#25) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Function DiamondCutFacet.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/utils/DiamondCutFacet.sol#22-60) contains magic numbers: 7, 3, 7, 3
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

DiamondLoupeFacet.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
```

```
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32=>uint256
Function DiamondLoupeFacet.facets() (contracts/utils/DiamondLoupeFacet.sol#26-76) has a dubious typecast: address=>bytes20
Function DiamondLoupeFacet.facetFunctionSelectors(address) (contracts/utils/DiamondLoupeFacet.sol#81-107) has a dubious typecast: address=>bytes20
Function DiamondLoupeFacet.facetAddresses() (contracts/utils/DiamondLoupeFacet.sol#111-145) has a dubious typecast: address=>bytes20
Function DiamondLoupeFacet.facetAddress(bytes4) (contracts/utils/DiamondLoupeFacet.sol#151-154) has a dubious typecast: address=>bytes20
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
DiamondLoupeFacet.facets() (contracts/utils/DiamondLoupeFacet.sol#26-76) uses assembly
        - INLINE ASM (contracts/utils/DiamondLoupeFacet.sol#68-70)
        - INLINE ASM (contracts/utils/DiamondLoupeFacet.sol#73-75)
DiamondLoupeFacet.facetFunctionSelectors(address) (contracts/utils/DiamondLoupeFacet.sol#81-107) uses assembly
        - INLINE ASM (contracts/utils/DiamondLoupeFacet.sol#104-106)
DiamondLoupeFacet.facetAddresses() (contracts/utils/DiamondLoupeFacet.sol#111-145) uses assembly
        - INLINE ASM (contracts/utils/DiamondLoupeFacet.sol#142-144)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/interfaces/IDiamondLoupe.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/DiamondLoupeFacet.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter DiamondLoupeFacet.facetFunctionSelectors(address)._facet (contracts/utils/DiamondLoupeFacet.sol#81) is not in mixedCase
Parameter DiamondLoupeFacet.facetAddress(bytes4)._functionSelector (contracts/utils/DiamondLoupeFacet.sol#151) is not in mixedCase
Parameter DiamondLoupeFacet.supportsInterface(bytes4)._interfaceId (contracts/utils/DiamondLoupeFacet.sol#157) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Variable DiamondLoupeFacet.facetFunctionSelectors(address)._facetFunctionSelectors (contracts/utils/DiamondLoupeFacet.sol#81) is too similar to IDiamondLoupe.facetFunctionSelectors(address).facetFunctionSelectors_ (contracts/interfaces/I
DiamondLoupe.sol#27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Function DiamondLoupeFacet.facets() (contracts/utils/DiamondLoupeFacet.sol#26-76) contains magic numbers: 8, 5, 255
Function DiamondLoupeFacet.facetFunctionSelectors(address) (contracts/utils/DiamondLoupeFacet.sol#81-107) contains magic numbers: 8, 5
Function DiamondLoupeFacet.facetAddresses() (contracts/utils/DiamondLoupeFacet.sol#111-145) contains magic numbers: 8, 5
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

EIP2535Initializable.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/EIP2535Initializable.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
```

```
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Initializer() (contracts/utils/EIP2535Initializable.sol#11-19) is not in mixedCase
Modifier EIP2535Initializable.EIP2535Reinitializer(uint8) (contracts/utils/EIP2535Initializable.sol#21-29) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

IrrigationAccessControl.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is never used and should be removed
IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is never used and should be removed
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.contractOwner() (contracts/libraries/LibDiamond.sol#49-51) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.enforceIsContractOwner() (contracts/libraries/LibDiamond.sol#53-55) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/IrrigationAccessControl.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init() (contracts/utils/IrrigationAccessControl.sol#13-16) is not in mixedCase
Function IrrigationAccessControl.__IrrigationAccessControl_init_unchained() (contracts/utils/IrrigationAccessControl.sol#18-22) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

OwnershipFacet.sol

```
INFO:Detectors:
Function LibDiamond.setContractOwner(address) (contracts/libraries/LibDiamond.sol#42-47) is a strange setter. Nothing is set in constructor or set in a function without using function parameters
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/strange_setter.md
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is an unprotected initializer.
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/unprotected_initialize.md
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex (contracts/libraries/LibDiamond.sol#118) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is a local variable never initialized
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).facetIndex (contracts/libraries/LibDiamond.sol#85) is a local variable never initialized
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains a low level call to a custom address
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/call_forward_to_protected.md
INFO:Detectors:
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: address<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: bytes32<=bytes20
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a dubious typecast: uint16<=uint256
Function LibDiamond.slitherConstructorConstantVariables() (contracts/libraries/LibDiamond.sol#13-244) has a dubious typecast: bytes32<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
```

```
INFO:Detectors:
LibDiamond.diamondStorage() (contracts/libraries/LibDiamond.sol#33-38) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#35-37)
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#227-230)
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) uses assembly
        - INLINE ASM (contracts/libraries/LibDiamond.sol#239-241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Different versions of Solidity are used:
        - Version used: ['^0.8.0', '^0.8.17']
        - ^0.8.0 (contracts/interfaces/IERC173Upgradeable.sol#2)
        - ^0.8.17 (contracts/interfaces/IDiamondCut.sol#2)
        - ^0.8.17 (contracts/libraries/LibDiamond.sol#2)
        - ^0.8.17 (contracts/utils/OwnershipFacet.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) has a high cyclomatic complexity (12).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) is never used and should be removed
LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) is never used and should be removed
LibDiamond.enforceHasContractCode(address,string) (contracts/libraries/LibDiamond.sol#237-243) is never used and should be removed
LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IDiamondCut.sol#2) allows old versions
Pragma version^0.8.0 (contracts/interfaces/IERC173Upgradeable.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/LibDiamond.sol#2) allows old versions
Pragma version^0.8.17 (contracts/utils/OwnershipFacet.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235):
        - (success,error) = _init.delegatecall(_calldata) (contracts/libraries/LibDiamond.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/libraries/LibDiamond.sol#42) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._diamondCut (contracts/libraries/LibDiamond.sol#69) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._init (contracts/libraries/LibDiamond.sol#70) is not in mixedCase
Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes)._calldata (contracts/libraries/LibDiamond.sol#71) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorCount (contracts/libraries/LibDiamond.sol#108) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectorSlot (contracts/libraries/LibDiamond.sol#109) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._newFacetAddress (contracts/libraries/LibDiamond.sol#110) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._action (contracts/libraries/LibDiamond.sol#111) is not in mixedCase
Parameter LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[])._selectors (contracts/libraries/LibDiamond.sol#112) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (contracts/libraries/LibDiamond.sol#217) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._contract (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter LibDiamond.enforceHasContractCode(address,string)._errorMessage (contracts/libraries/LibDiamond.sol#237) is not in mixedCase
Parameter OwnershipFacet.transferOwnership(address)._newOwner (contracts/utils/OwnershipFacet.sol#8) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_2 (contracts/libraries/LibDiamond.sol#141) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).oldFacet_scope_5 (contracts/libraries/LibDiamond.sol#171)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_0 (contracts/libraries/LibDiamond.sol#139) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(ui
nt256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selectorIndex_scope_3 (contracts/libraries/LibDiamond.sol#156)
Variable LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_1 (contracts/libraries/LibDiamond.sol#140) is too similar to LibDiamond.addReplaceRemoveFacetSelectors(uint256
,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]).selector_scope_4 (contracts/libraries/LibDiamond.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
Function LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes) (contracts/libraries/LibDiamond.sol#68-105) contains magic numbers: 7, 3, 7, 3
Function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) contains magic numbers: 7, 224, 3, 3, 7, 7, 5, 3, 7, 8
Function LibDiamond.initializeDiamondCut(address,bytes) (contracts/libraries/LibDiamond.sol#217-235) contains magic number: 32
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
INFO:Detectors:
In a function LibDiamond.addReplaceRemoveFacetSelectors(uint256,bytes32,address,IDiamondCut.FacetCutAction,bytes4[]) (contracts/libraries/LibDiamond.sol#107-215) variable LibDiamond.CLEAR_SELECTOR_MASK (contracts/libraries/LibDiamond.sol
#60) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

PodTransferHelper.sol

```
INFO:Detectors:
PodTransferHelper.getPlotSplittedByFM(uint256,uint256,uint256,uint256) (contracts/libraries/PodTransferHelper.sol#9-36) is never used and should be removed
PodTransferHelper.getPlotWithOffset(uint256,uint256,uint256,uint256,uint256,uint128[6]) (contracts/libraries/PodTransferHelper.sol#39-72) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/libraries/PodTransferHelper.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function PodTransferHelper.getPlotWithOffset(uint256,uint256,uint256,uint256,uint256,uint128[6]) (contracts/libraries/PodTransferHelper.sol#39-72) contains magic number: 3
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
```

LibPrice.sol

```
INFO:Detectors:
LibPrice.getPriceOfPods(uint256,uint256,uint256,uint256) (contracts/libraries/Oracle/LibPrice.sol#19-36) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.17 (contracts/libraries/Oracle/LibPrice.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
In a function LibPrice.getPriceOfPods(uint256,uint256,uint256,uint256) (contracts/libraries/Oracle/LibPrice.sol#19-36) variable LibPrice.BDV_FACTOR (contracts/libraries/Oracle/LibPrice.sol#7) is read multiple times
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/multiple_storage_read.md
```

BeanPriceOracle.sol

```
INFO:Detectors:
Different versions of Solidity are used:
        - Version used: ['0.8.17', '^0.8.17']
        - 0.8.17 (contracts/interfaces/ICustomOracle.sol#2)
        - 0.8.17 (contracts/oracles/BeanPriceOracle.sol#2)
        - ^0.8.17 (contracts/interfaces/IBeanstalkPrice.sol#2)
        - ^0.8.17 (contracts/libraries/Constants.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
INFO:Detectors:
Pragma version^0.8.17 (contracts/interfaces/IBeanstalkPrice.sol#2) allows old versions
Pragma version0.8.17 (contracts/interfaces/ICustomOracle.sol#2) allows old versions
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version0.8.17 (contracts/oracles/BeanPriceOracle.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function BeanPriceOracle.latestPrice() (contracts/oracles/BeanPriceOracle.sol#14-16) contains magic number: 1e12
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
```

ChainlinkOracle.sol

```
INFO:Detectors:
ChainlinkOracle.setAdmin(address).newAdmin (contracts/oracles/ChainlinkOracle.sol#60) lacks a zero-check on :
        - admin = newAdmin (contracts/oracles/ChainlinkOracle.sol#62)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
ChainlinkOracle.getChainlinkPrice(AggregatorV2V3Interface) (contracts/oracles/ChainlinkOracle.sol#30-39) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/libraries/Constants.sol#2) allows old versions
Pragma version^0.8.17 (contracts/oracles/AggregatorV2V3Interface.sol#3) allows old versions
Pragma version^0.8.17 (contracts/oracles/ChainlinkOracle.sol#3) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function ChainlinkOracle.getChainlinkPrice(AggregatorV2V3Interface) (contracts/oracles/ChainlinkOracle.sol#30-39) contains magic numbers: 18, 10
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/magic_number.md
```

UniswapV3Twap.sol

```
INFO:Detectors:
Function UniswapV3Twap.getSqrtTwapX96(address,uint32) (contracts/oracles/uniswapv3/UniswapV3Twap.sol#11-30) has a dubious typecast: int56<=int32
Function UniswapV3Twap.getTwap(uint256,address,address,uint32) (contracts/oracles/uniswapv3/UniswapV3Twap.sol#32-47) has a dubious typecast: uint256<=uint256
Reference: https://github.com/pessimistic-io/slitherin/blob/master/docs/dubious_typecast.md
INFO:Detectors:
Pragma version^0.8.0 (contracts/oracles/uniswapv3/UniswapV3Twap.sol#2) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
UniswapV3Twap.slitherConstructorConstantVariables() (contracts/oracles/uniswapv3/UniswapV3Twap.sol#8-48) uses literals with too many digits:
        - Q192 = 0x1000000000000000000000000000000000000000000000000 (contracts/oracles/uniswapv3/UniswapV3Twap.sol#9)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

- The unprotected initialize issues flagged by Slither were checked individually and are false positives.
- No major issues found by Slither.

6.2 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues and to identify low-hanging fruits on the targets for this engagement. Among the tools used was MythX, a security analysis service for Ethereum smart contracts. MythX performed a scan on the smart contracts and sent the compiled results to the analyzers to locate any vulnerabilities.

MythX results:

SprinklerUpgradeable.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
17	(SWC-123) Requirement Violation	Low	Requirement violation.
139	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
156	(SWC-123) Requirement Violation	Low	Requirement violation.

WaterTowerUpgradeable.sol

Line	SWC Title	Severity	Short Description
27	(SWC-123) Requirement Violation	Low	Requirement violation.
67	(SWC-107) Reentrancy	Low	A call to a user-supplied address is executed.

TrancheBondUpgradeable.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
27	(SWC-123) Requirement Violation	Low	Requirement violation.
289	(SWC-123) Requirement Violation	Low	Requirement violation.

AuctionUpgradeable.sol

Line	SWC Title	Severity	Short Description
29	(SWC-123) Requirement Violation	Low	Requirement violation.
123	(SWC-123) Requirement Violation	Low	Requirement violation.
143	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
143	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
172	(SWC-107) Reentrancy	Low	Write to persistent state following external call.
172	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
186	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
188	(SWC-107) Reentrancy	Low	Read of persistent state following external call.

ERC1155WhitelistUpgradeable.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

WaterCommonUpgradeable.sol

No issues found by MythX.

PriceOracleUpgradeable.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
14	(SWC-123) Requirement Violation	Low	Requirement violation.
23	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
24	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
87	(SWC-123) Requirement Violation	Low	Requirement violation.

PodsOracleUpgradeable.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

WaterUpgradeable.sol

No issues found by MythX.

IrrigationDiamond.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

Diamond.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
21	(SWC-110) Assert Violation	Unknown	Out of bounds array access
22	(SWC-110) Assert Violation	Unknown	Out of bounds array access
33	(SWC-109) Uninitialized Storage Pointer	Medium	Dangerous use of uninitialized storage variables.

DiamondCutFacet.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.
15	(SWC-123) Requirement Violation	Low	Requirement violation.

DiamondLoupeFacet.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

EIP2535Initializable.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

IrrigationAccessControl.sol

Line	SWC Title	Severity	Short Description
3	(SWC-103) Floating Pragma	Low	A floating pragma is set.

OwnershipFacet.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

PodTransferHelper.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

LibPrice.sol

No issues found by MythX.

BeanPriceOracle.sol

No issues found by MythX.

ChainlinkOracle.sol

Line	SWC Title	Severity	Short Description
3	(SWC-103) Floating Pragma	Low	A floating pragma is set.

UniswapV3Twap.sol

Line	SWC Title	Severity	Short Description
2	(SWC-103) Floating Pragma	Low	A floating pragma is set.

- MythX flagged some integer overflows and underflows which all were false positives, as the contracts are using Solidity ^0.8.17 version. After the Solidity version 0.8.0 Arithmetic operations revert to underflow and overflow by default.
- MythX also flagged some assert violations, which were all considered to be false positives.
- No major issues were found by MythX.

THANK YOU FOR CHOOSING

// HALBORN