



# DeBridge - Solidity

## Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **May 25th, 2022 - June 15th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY	3
CONTACTS	3
1 EXECUTIVE OVERVIEW	4
1.1 INTRODUCTION	5
1.2 AUDIT SUMMARY	5
1.3 TEST APPROACH & METHODOLOGY	5
RISK METHODOLOGY	6
1.4 SCOPE	8
2 ASSESSMENT SUMMARY & FINDINGS OVERVIEW	9
3 FINDINGS & TECH DETAILS	10
3.1 (HAL-01) ZERO ADDRESS NOT CHECKED - INFORMATIONAL	12
Description	12
Code Location	12
Risk Level	16
Recommendation	16
Remediation Plan	16
3.2 (HAL-02) USE ++I INSTEAD OF I++ IN LOOPS FOR GAS OPTIMIZATION - INFORMATIONAL	17
Description	17
Code Location	17
Risk Level	24
Recommendation	24
Remediation Plan	25
4 AUTOMATED TESTING	26
4.1 STATIC ANALYSIS REPORT	27

Description	27
Slither results	27
4.2 AUTOMATED SECURITY SCAN	45
Description	45
MythX results	45

## DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	06/14/2022	Omar Alshaeb
0.2	Draft Review	06/16/2022	Gabi Urrutia
1.0	Remediation Plan	06/16/2022	Omar Alshaeb
1.1	Remediation Plan Review	06/16/2022	Gabi Urrutia

## 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
Omar Alshaeb	Halborn	Omar.Alshaeb@halborn.com

# EXECUTIVE OVERVIEW

## 1.1 INTRODUCTION

DeBridge engaged Halborn to conduct a security audit on their smart contracts beginning on May 25th, 2022 and ending on June 15th, 2022 . The security assessment was scoped to the smart contracts provided to the Halborn team.

## 1.2 AUDIT SUMMARY

The team at Halborn was provided three weeks for the engagement and assigned a full-time security engineer to audit the security of the smart contract. 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 audit is to:

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

In summary, Halborn did not identify security risks, but just some suggestions that were addressed by the DeBridge 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 audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- 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 hotspots or bugs. ([MythX](#))
- Static Analysis of security for scoped contract, and imported functions. ([Slither](#))
- Testnet deployment ([Brownie](#), [Remix IDE](#))

#### RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

#### RISK SCALE - LIKELIHOOD

- 5 - Almost certain an incident will occur.
- 4 - High probability of an incident occurring.
- 3 - Potential of a security incident in the long term.
- 2 - Low probability of an incident occurring.
- 1 - Very unlikely issue will cause an incident.

#### RISK SCALE - IMPACT

- 5 - May cause devastating and unrecoverable impact or loss.
- 4 - May cause a significant level of impact or loss.

- 3 - May cause a partial impact or loss to many.
- 2 - May cause temporary impact or loss.
- 1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of **10** to **1** with **10** being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
----------	------	--------	-----	---------------

- 10** - CRITICAL
- 9** - **8** - HIGH
- 7** - **6** - MEDIUM
- 5** - **4** - LOW
- 3** - **1** - VERY LOW AND INFORMATIONAL

## 1.4 SCOPE

### IN-SCOPE:

The security assessment was scoped to the following smart contracts:

- DeBridgeGate.sol
- DeBridgeTokenDeployer.sol
- OraclesManager.sol
- SignatureVerifier.sol
- WethGate.sol
- SimpleFeeProxy.sol
- DefiController.sol
- CallProxy.sol
- DeBridgeToken.sol
- FeeProxy.sol
- Claimer.sol
- MultiSendCallOnly.sol

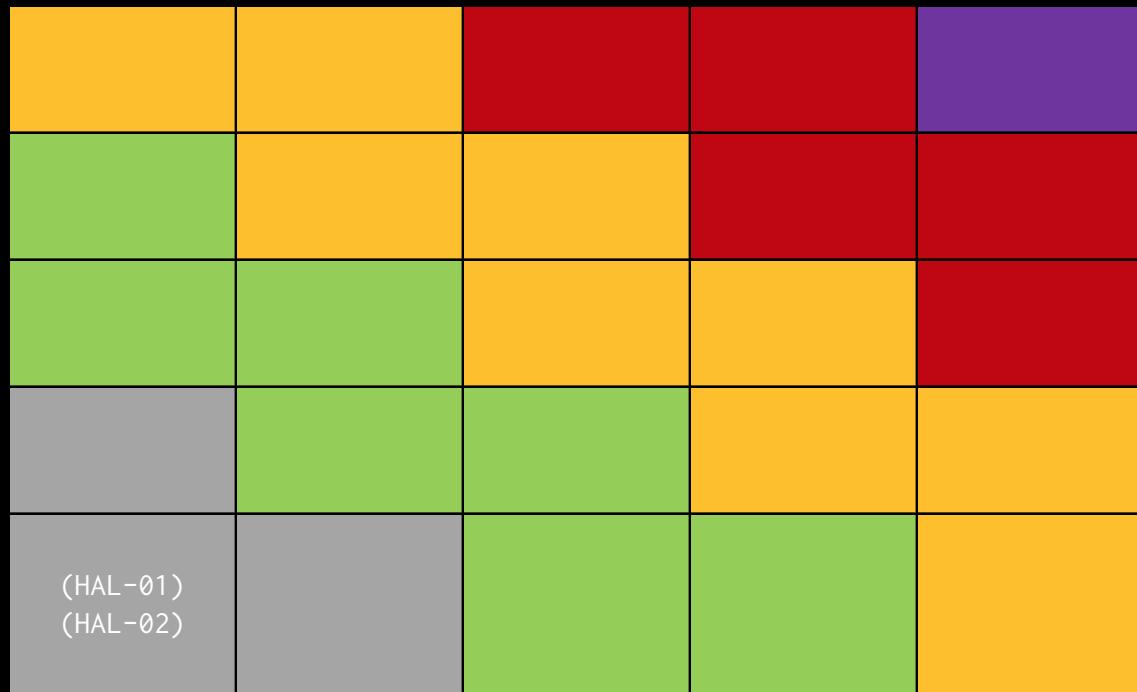
Commit ID: 47335f5f8f46ccd52594d594cb1d2c7a1d6dcf2c

## 2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	0	2

LIKELIHOOD

IMPACT



# EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - ZERO ADDRESS NOT CHECKED	Informational	ACKNOWLEDGED
HAL02 - USE ++I INSTEAD OF I++ IN LOOPS FOR GAS OPTIMIZATION	Informational	ACKNOWLEDGED



# FINDINGS & TECH DETAILS



### 3.1 (HAL-01) ZERO ADDRESS NOT CHECKED - INFORMATIONAL

Description:

In some blocks of code, the address variables are not being checked to avoid pointing to the zero address.

Code Location:

Listing 1: DeBridgeGate.sol (Line 173)

```
168 function initialize(
169     uint8 _excessConfirmations,
170     IWETH _weth
171 ) public initializer {
172     excessConfirmations = _excessConfirmations;
173     weth = _weth;
174
175     _setupRole(DEFAULT_ADMIN_ROLE, msg.sender);
176     __ReentrancyGuard_init();
177 }
```

Listing 2: DeBridgeGate.sol (Line 424)

```
423 function setCallProxy(address _callProxy) external onlyAdmin {
424     callProxy = _callProxy;
425     emit CallProxyUpdated(_callProxy);
426 }
```

Listing 3: DeBridgeGate.sol (Line 451)

```
450 function setSignatureVerifier(address _verifier) external
↳ onlyAdmin {
451     signatureVerifier = _verifier;
452 }
```

**Listing 4: DeBridgeGate.sol (Line 457)**

```

456 function setDeBridgeTokenDeployer(address _deBridgeTokenDeployer)
↳ external onlyAdmin {
457     deBridgeTokenDeployer = _deBridgeTokenDeployer;
458 }
```

**Listing 5: DeBridgeGate.sol (Line 463)**

```

462 function setDefiController(address _defiController) external
↳ onlyAdmin {
463     defiController = _defiController;
464 }
```

**Listing 6: DeBridgeGate.sol (Line 469)**

```

468 function setFeeContractUpdater(address _value) external onlyAdmin
↳ {
469     feeContractUpdater = _value;
470 }
```

**Listing 7: DeBridgeGate.sol (Line 475)**

```

474 function setWethGate(IWethGate _wethGate) external onlyAdmin {
475     wethGate = _wethGate;
476 }
```

**Listing 8: DeBridgeGate.sol (Line 548)**

```

547 function setFeeProxy(address _feeProxy) external onlyAdmin {
548     feeProxy = _feeProxy;
549 }
```

**Listing 9: DeBridgeTokenDeployer.sol (Lines 73,74,75)**

```

68 function initialize(
69     address _tokenImplementation,
70     address _deBridgeTokenAdmin,
71     address _debridgeAddress
72 ) public initializer {
```

```

73     tokenImplementation = _tokenImplementation;
74     deBridgeTokenAdmin = _deBridgeTokenAdmin;
75     debridgeAddress = _debridgeAddress;
76
77     _setupRole(DEFAULT_ADMIN_ROLE, msg.sender);
78 }
```

**Listing 10:** SignatureVerifier.sol (Line 53)

```

45 function initialize(
46     uint8 _minConfirmations,
47     uint8 _confirmationThreshold,
48     uint8 _excessConfirmations,
49     address _debridgeAddress
50 ) public initializer {
51     OraclesManager.initialize(_minConfirmations,
52     ↳ _excessConfirmations);
52     confirmationThreshold = _confirmationThreshold;
53     debridgeAddress = _debridgeAddress;
54 }
```

**Listing 11:** SignatureVerifier.sol (Line 128)

```

127 function setDebridgeAddress(address _debridgeAddress) external
128     ↳ onlyAdmin {
128     debridgeAddress = _debridgeAddress;
129 }
```

**Listing 12:** WethGate.sol (Line 27)

```

26 constructor(IWETH _weth) {
27     weth = _weth;
28 }
```

**Listing 13:** SimpleFeeProxy.sol (Lines 37,38)

```

36 function initialize(IDeBridgeGate _debridgeGate, address _treasury
37     ↳ ) public initializer {
37     debridgeGate = _debridgeGate;
38     treasury = _treasury;
39     _setupRole(DEFAULT_ADMIN_ROLE, msg.sender);
```

```
40 }
```

**Listing 14: SimpleFeeProxy.sol (Line 53)**

```
52 function setDebridgeGate(IDeBridgeGate _debridgeGate) external  
↳ onlyAdmin {  
53     debridgeGate = _debridgeGate;  
54 }
```

**Listing 15: SimpleFeeProxy.sol (Line 57)**

```
56 function setTreasury(address _treasury) external onlyAdmin {  
57     treasury = _treasury;  
58 }
```

**Listing 16: DefiController.sol (Line 266)**

```
265 function setDeBridgeGate(IDeBridgeGate _deBridgeGate) external  
↳ onlyAdmin {  
266     deBridgeGate = _deBridgeGate;  
267 }
```

**Listing 17: FeeProxy.sol (Lines 63,64)**

```
62 function initialize(IUniswapV2Factory _uniswapFactory, IWETH _weth  
↳ ) public initializer {  
63     uniswapFactory = _uniswapFactory;  
64     weth = _weth;  
65     _setupRole(DEFAULT_ADMIN_ROLE, msg.sender);  
66 }
```

**Listing 18: FeeProxy.sol (Line 79)**

```
78 function setUniswapFactory(IUniswapV2Factory _uniswapFactory)  
↳ external onlyAdmin {  
79     uniswapFactory = _uniswapFactory;  
80 }
```

**Listing 19: FeeProxy.sol (Line 83)**

```
82 function setDebridgeGate(IDeBridgeGate _debridgeGate) external
↳ onlyAdmin {
83     debridgeGate = _debridgeGate;
84 }
```

**Listing 20: Claimer.sol (Line 63)**

```
58 function initialize(
59     DeBridgeGate _deBridgeGate
60 ) public initializer {
61     _setupRole(DEFAULT_ADMIN_ROLE, msg.sender);
62
63     deBridgeGate = _deBridgeGate;
64 }
```

**Listing 21: Claimer.sol (Line 159)**

```
158 function setDeBridgeGate(DeBridgeGate _deBridgeGate) external
↳ onlyAdmin {
159     deBridgeGate = _deBridgeGate;
160 }
```

Risk Level:

**Likelihood** - 1

**Impact** - 1

Recommendation:

When setting an address variable, always make sure the value is not zero.

Remediation Plan:

**ACKNOWLEDGED:** The DeBridge team acknowledged this issue.

## 3.2 (HAL-02) USE `++i` INSTEAD OF `i++` IN LOOPS FOR GAS OPTIMIZATION - INFORMATIONAL

### Description:

In some loops, the variable `i` is incremented using `i++`. It is known that, in loops, using `++i` costs less gas per iteration than `i++`. This also affects variables incremented inside the loop code block.

### Code Location:

```
Listing 22: DeBridgeGate.sol (Line 361)

355 function updateChainSupport(
356     uint256[] memory _chainIds,
357     ChainSupportInfo[] memory _chainSupportInfo,
358     bool _isChainFrom
359 ) external onlyAdmin {
360     if (_chainIds.length != _chainSupportInfo.length) revert
↳ WrongArgument();
361     for (uint256 i = 0; i < _chainIds.length; i++) {
362         if(_isChainFrom){
363             getChainFromConfig[_chainIds[i]] = _chainSupportInfo[i]
↳ ];
364         }
365         else {
366             getChainToConfig[_chainIds[i]] = _chainSupportInfo[i];
367         }
368         emit ChainsSupportUpdated(_chainIds[i], _chainSupportInfo[
↳ i], _isChainFrom);
369     }
370 }
```

**Listing 23:** DeBridgeGate.sol (Line 395)

```

388 function updateAssetFixedFees(
389     bytes32 _debridgeId,
390     uint256[] memory _supportedChainIds,
391     uint256[] memory _assetFeesInfo
392 ) external onlyAdmin {
393     if (_supportedChainIds.length != _assetFeesInfo.length) revert
394     WrongArgument();
395     DebridgeFeeInfo storage debridgeFee = getDebridgeFeeInfo[
396     _debridgeId];
397     for (uint256 i = 0; i < _supportedChainIds.length; i++) {
398         debridgeFee.getChainFee[_supportedChainIds[i]] =
399         _assetFeesInfo[i];
400     }
401 }
```

**Listing 24:** DeBridgeGate.sol (Line 555)

```

554 function blockSubmission(bytes32[] memory _submissionIds, bool
555     isBlocked) external onlyAdmin {
556     for (uint256 i = 0; i < _submissionIds.length; i++) {
557         isBlockedSubmission[_submissionIds[i]] = isBlocked;
558         if (isBlocked) {
559             emit Blocked(_submissionIds[i]);
560         } else {
561             emit Unblocked(_submissionIds[i]);
562         }
563     }
564 }
```

**Listing 25:** DeBridgeTokenDeployer.sol (Line 177)

```

172 function setOverridedTokenInfo (
173     bytes32[] memory _debridgeIds,
174     OverridedTokenInfo[] memory _tokens
175 ) external onlyAdmin {
176     if (_debridgeIds.length != _tokens.length) revert
177     WrongArgument();
178     for (uint256 i = 0; i < _debridgeIds.length; i++) {
179         overridedTokens[_debridgeIds[i]] = _tokens[i];
```

```

179     }
180 }
181

```

**Listing 26: OraclesManager.sol (Line 85)**

```

78 function addOracles(
79     address[] memory _oracles,
80     bool[] memory _required
81 ) external onlyAdmin {
82     if (_oracles.length != _required.length) revert WrongArgument
83     if (minConfirmations < (oracleAddresses.length + _oracles.
84     length) / 2 + 1) revert LowMinConfirmations();
85     for (uint256 i = 0; i < _oracles.length; i++) {
86         OracleInfo storage oracleInfo = getOracleInfo[_oracles[i]
87     }];
88         if (oracleInfo.exist) revert OracleAlreadyExist();
89         oracleAddresses.push(_oracles[i]);
90
91         if (_required[i]) {
92             requiredOraclesCount += 1;
93         }
94
95         oracleInfo.exist = true;
96         oracleInfo.isValid = true;
97         oracleInfo.required = _required[i];
98
99         emit AddOracle(_oracles[i], _required[i]);
100    }
101 }
102

```

**Listing 27: OraclesManager.sol (Line 125)**

```

107 function updateOracle(
108     address _oracle,
109     bool _isValid,
110     bool _required
111 ) external onlyAdmin {
112     //If oracle is invalid, it must be not required

```

```

113     if (!isValid && _required) revert WrongArgument();
114
115     OracleInfo storage oracleInfo = getOracleInfo[_oracle];
116     if (!oracleInfo.exist) revert OracleNotFound();
117
118     if (oracleInfo.required && !_required) {
119         requiredOraclesCount -= 1;
120     } else if (!_oracleInfo.required && _required) {
121         requiredOraclesCount += 1;
122     }
123     if (oracleInfo.isValid && !_isValid) {
124         // remove oracle from oracleAddresses array without
125         // keeping an order
126         for (uint256 i = 0; i < oracleAddresses.length; i++) {
127             if (oracleAddresses[i] == _oracle) {
128                 oracleAddresses[i] = oracleAddresses[
129                 oracleAddresses.length - 1];
130                 oracleAddresses.pop();
131                 break;
132             }
133         } else if (!_oracleInfo.isValid && _isValid) {
134             if (minConfirmations < (oracleAddresses.length + 1) / 2 +
135             1) revert LowMinConfirmations();
136             oracleAddresses.push(_oracle);
137         }
138         oracleInfo.isValid = _isValid;
139         oracleInfo.required = _required;
140         emit UpdateOracle(_oracle, _required, _isValid);
141     }

```

Listing 28: SignatureVerifier.sol (Lines 73,77)

```

58 function submit(
59     bytes32 _submissionId,
60     bytes memory _signatures,
61     uint8 _excessConfirmations
62 ) external override onlyDeBridgeGate {
63     //Need confirmation to confirm submission
64     uint8 needConfirmations = _excessConfirmations >
65     minConfirmations
66     ? _excessConfirmations
67     : minConfirmations;

```

```
67     // Count of required(DSRM) oracles confirmation
68     uint256 currentRequiredOraclesCount;
69     // stack variable to aggregate confirmations and write to
70     // storage once
71     uint8 confirmations;
72     uint256 signaturesCount = _countSignatures(_signatures);
73     address[] memory validators = new address[](signaturesCount);
74     for (uint256 i = 0; i < signaturesCount; i++) {
75         (bytes32 r, bytes32 s, uint8 v) = _signatures.
76         parseSignature(i * 65);
77         address oracle = ecrecover(_submissionId.getUnsignedMsg(),
78         v, r, s);
79         if (getOracleInfo[oracle].isValid) {
80             for (uint256 k = 0; k < i; k++) {
81                 if (validators[k] == oracle) revert
82                 DuplicateSignatures();
83             }
84             validators[i] = oracle;
85             confirmations += 1;
86             emit Confirmed(_submissionId, oracle);
87             if (getOracleInfo[oracle].required) {
88                 currentRequiredOraclesCount += 1;
89             }
90             if (
91                 confirmations >= needConfirmations &&
92                 currentRequiredOraclesCount >=
93                 requiredOraclesCount
94             ) {
95                 break;
96             }
97         }
98     }
99     if (currentRequiredOraclesCount != requiredOraclesCount)
100         revert NotConfirmedByRequiredOracles();
101     if (confirmations >= minConfirmations) {
102         if (currentBlock == uint40(block.number)) {
103             submissionsInBlock += 1;
104         } else {
105             currentBlock = uint40(block.number);
106             submissionsInBlock = 1;
107         }
108     }
109 }
```

```
106         emit SubmissionApproved(_submissionId);
107     }
108
109     if (submissionsInBlock > confirmationThreshold) {
110         if (confirmations < excessConfirmations) revert
111         ↳ NotConfirmedThreshold();
112     }
113     if (confirmations < needConfirmations) revert
114         ↳ SubmissionNotConfirmed();
115 }
```

Listing 29: DeBridgeToken.sol (Line 71)

```
54 function initialize(
55     string memory name_,
56     string memory symbol_,
57     uint8 decimals_,
58     address admin,
59     address[] memory minters
60 ) public initializer {
61     _decimals = decimals_;
62     name_ = string(abi.encodePacked("deBridge ",
63         bytes(name_).length == 0 ? symbol_ : name_));
64     symbol_ = string(abi.encodePacked("de", symbol_));
65
66     __ERC20_init_unchained(name_, symbol_);
67
68     _setupRole(DEFAULT_ADMIN_ROLE, admin);
69     _setupRole(PAUSER_ROLE, admin);
70     uint256 mintersCount = minters.length;
71     for (uint256 i = 0; i < mintersCount; i++) {
72         _setupRole(MINTER_ROLE, minters[i]);
73     }
74
75     uint256 chainId;
76     assembly {
77         chainId := chainid()
78     }
79     DOMAIN_SEPARATOR = keccak256(
80         abi.encode(
81             keccak256(
```

```

82             "EIP712Domain(string name,string version,uint256
83             chainId,address verifyingContract)"
84             ),
85             keccak256(bytes(name_)),
86             keccak256(bytes("1")),
87             chainId,
88             address(this)
89         )
90     );
91 }
```

Listing 30: Claimer.sol (Line 70)

```

66 function batchClaim(
67     ClaimInfo[] calldata _claims
68 ) external {
69     uint256 claimsCount = _claims.length;
70     for (uint256 i = 0; i < claimsCount; i++) {
71         ClaimInfo memory claim = _claims[i];
72         try deBridgeGate.claim(
73             claim.debridgeId,
74             claim.amount,
75             claim.chainIdFrom,
76             claim.receiver,
77             claim.nonce,
78             claim.signatures,
79             claim.autoParams)
80         {} 
81         catch {
82             emit BatchError(i);
83         }
84     }
85 }
```

Listing 31: Claimer.sol (Line 91)

```

87 function batchAssetsDeploy(
88     AssetDeployInfo[] calldata _deploys
89 ) external {
90     uint256 count = _deploys.length;
91     for (uint256 i = 0; i < count; i++) {
```

```
92     AssetDeployInfo memory deploy = _deploys[i];
93     try deBridgeGate.deployNewAsset(
94         deploy.nativeTokenAddress,
95         deploy.nativeChainId,
96         deploy.name,
97         deploy.symbol,
98         deploy.decimals,
99         deploy.signatures)
100    { }
101   catch {
102       emit BatchError(i);
103   }
104 }
105 }
```

**Listing 32: Claimer.sol (Line 143)**

```
141 function withdrawFee(address[] memory _tokenAddresses) external
142     onlyAdmin {
143     uint256 lenght = _tokenAddresses.length;
144     for (uint i = 0; i < lenght; i++) {
145         IERC20(_tokenAddresses[i]).transfer(
146             msg.sender,
147             IERC20(_tokenAddresses[i]).balanceOf(address(this))
148         );
149     }
150 }
```

Risk Level:

**Likelihood - 1**

**Impact - 1**

Recommendation:

It is recommended to use `++i` instead of `i++` to increment the value of a `uint` variable inside a loop. This also applies to variables declared inside the `for` loop, but does not apply outside of loops.

## FINDINGS & TECH DETAILS

Remediation Plan:

**ACKNOWLEDGED:** The DeBridge team acknowledged this issue.

# AUTOMATED TESTING

## 4.1 STATIC ANALYSIS REPORT

### Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contracts. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified all the contracts in the repository and was able to compile them correctly into their ABI and binary formats, Slither was run on the all-scoped 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:

#### DeBridgeGate.sol

```

Reentrancy in DeBridgeGate._send(bytes,address,uint256,uint256,bool) (contracts/transfers/DeBridgeGate.sol#659-884)
  External calls:
    - _validateToken(tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
      - (success) = token.call(tokenAddress,abi.encodeWithSignature("decimals()")) (contracts/transfers/DeBridgeGate.sol#885)
      - (success,None) = token.call(tokenAddress,abi.encodeWithSignature("symbol()")) (contracts/transfers/DeBridgeGate.sol#889)
  - IERC20Permit(_tokenAddress).permit(msg.sender,address(this),_amount,deadline,v,r,s) (contracts/transfers/DeBridgeGate.sol#677-684)
  - weth.deposit(value,_amount)() (contracts/transfers/DeBridgeGate.sol#872)
  - token.transferFrom(deployer,DeBridgeGate,amount) (contracts/transfers/DeBridgeGate.sol#731)
  - _safeTransferETHmsg.sender,msg.value-nativeFee (contracts/transfers/DeBridgeGate.sol#749)
    - (success) = to.call(value:value,in(n bytes)) (contracts/transfers/DeBridgeGate.sol#980)
  External calls sending eth:
  - weth.depositNative() (contracts/transfers/DeBridgeGate.sol#726)
  - _safeTransferETHmsg.sender,msg.value - nativeFee (contracts/transfers/DeBridgeGate.sol#749)
    - (success) = to.call(value:value,in(n bytes)) (contracts/transfers/DeBridgeGate.sol#989)
  State variables written after the call(s):
  - _debridgeBalance += amount + nativeFee (contracts/transfers/DeBridgeGate.sol#777)
  - _debridgeBalance -= amount + nativeFee (contracts/transfers/DeBridgeGate.sol#948)
  - getDebridgeFeeInfo(nativeBridgeId).collectedFee += nativeFee (contracts/transfers/DeBridgeGate.sol#772)
  - debridgeFee.collectedFees += totalFee (contracts/transfers/DeBridgeGate.sol#786)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) shadows:
  - ERC165Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36)

ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#57) shadows:
  - PausableUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
    - ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

DeBridgeGate._normalizeTokenAmount(address,uint256) (contracts/transfers/DeBridgeGate.sol#181-1924) performs a multiplication on the result of a division:
  - _amount = _amount / multiple * multiplier (contracts/transfers/DeBridgeGate.sol#1021)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint256,bytes) (contracts/transfers/DeBridgeGate.sol#182-239) uses a dangerous strict equality:
  - autoParams.data.length > 0 && autoParams.fallbackAddress.length == 0 (contracts/transfers/DeBridgeGate.sol#211)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in DeBridgeGate.deployNAsset(bytes,uint256,string,string,uint8,bytes) (contracts/transfers/DeBridgeGate.sol#317-338):
  External calls:
  - ISignatureVerifier(signatureVerifier).submit(deployId,_signatures,excessConfirmations) (contracts/transfers/DeBridgeGate.sol#332)
  - DeBridgeGate._deployNAsset(IDeBridgeGateDeployer(deBridgeTokenDeployer).deployAsset(deBridgeId,_name,_symbol,_decimals)) (contracts/transfers/DeBridgeGate.sol#334-335)
  State variables written after the call(s):
  - _addAsset(deBridgeId,tokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
    - debridge.exist = true (contracts/transfers/DeBridgeGate.sol#628)
    - debridge.tokenAddress = tokenAddress (contracts/transfers/DeBridgeGate.sol#629)
    - debridge.nativeTokenAddress = _nativeTokenAddress (contracts/transfers/DeBridgeGate.sol#630)
    - debridge.maxAmount = type(uint256).max (contracts/transfers/DeBridgeGate.sol#633)
    - debridge.minReservesBps = uint16(BPS_DENOMINATOR) (contracts/transfers/DeBridgeGate.sol#636)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

DeBridgeGate.claim(bytes32,uint256,uint256,uint256,bytes,bytes),autoParams (contracts/transfers/DeBridgeGate.sol#243) is a local variable never initialized
DeBridgeGate._send(bytes,address,uint256,uint256,bool).assetsFixedFee (contracts/transfers/DeBridgeGate.sol#744) is a local variable never initialized
DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint256,bytes),autoParams (contracts/transfers/DeBridgeGate.sol#204) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

DeBridgeGate.setDefController(address) (contracts/transfers/DeBridgeGate.sol#462-464) should emit an event for:
  - _defController = _defController (contracts/transfers/DeBridgeGate.sol#463)
DeBridgeGate.setFeeProxy(address) (contracts/transfers/DeBridgeGate.sol#468-478) should emit an event for:
  - _feeProxy = _feeProxy (contracts/transfers/DeBridgeGate.sol#548)
DeBridgeGate.setFeeProxy(address) (contracts/transfers/DeBridgeGate.sol#567-569) should emit an event for:
  - _feeProxy = _feeProxy (contracts/transfers/DeBridgeGate.sol#568)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

DeBridgeGate.initialize(uint8,IERC20) (contracts/transfers/DeBridgeGate.sol#162-177) should emit an event for:
  - excessConfirmations = excessConfirmations (contracts/transfers/DeBridgeGate.sol#172)
DeBridgeGate.updateExcessConfirmations(uint) (contracts/transfers/DeBridgeGate.sol#446-450) should emit an event for:
  - flashFeeBps = flashFeeBps (contracts/transfers/DeBridgeGate.sol#446)
DeBridgeGate.updateFlashFee(uint256) (contracts/transfers/DeBridgeGate.sol#567-579) should emit an event for:
  - flashFeeBps = _flashFeeBps (contracts/transfers/DeBridgeGate.sol#569)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

DeBridgeGate.setCallProxy(address).callProxy (contracts/transfers/DeBridgeGate.sol#423) lacks a zero-check on :

```



# AUTOMATED TESTING

# AUTOMATED TESTING

```

Parameter DeBridgeGate.getSubmissionIdFrom(bytes32, uint256, uint256, address, uint256, IDEBridgeGate.SubmissionAutoParamsFrom, bool, address), _amount (contracts/transfers/DeBridgeGate.sol#1074) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32, uint256, uint256, address, uint256, IDEBridgeGate.SubmissionAutoParamsFrom, bool, address), _receiver (contracts/transfers/DeBridgeGate.sol#1075) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32, uint256, uint256, address, uint256, IDEBridgeGate.SubmissionAutoParamsFrom, bool, address), _nonce (contracts/transfers/DeBridgeGate.sol#1076) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32, uint256, uint256, address, uint256, IDEBridgeGate.SubmissionAutoParamsFrom, bool, address), _autoParam (contracts/transfers/DeBridgeGate.sol#1077) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32, uint256, uint256, address, uint256, IDEBridgeGate.SubmissionAutoParamsFrom, bool, address), _sender (contracts/transfers/DeBridgeGate.sol#1078) is not in mixedCase
Parameter DeBridgeGate.getDeployId(bytes32, string, string, uint8), _debridgeId (contracts/transfers/DeBridgeGate.sol#1120) is not in mixedCase
Parameter DeBridgeGate.getDeployId(bytes32, string, string, uint8), _name (contracts/transfers/DeBridgeGate.sol#1121) is not in mixedCase
Parameter DeBridgeGate.getDeployId(bytes32, string, string, uint8), _decimals (contracts/transfers/DeBridgeGate.sol#1122) is not in mixedCase
Parameter DeBridgeGate.getDeployId(bytes32, string, string, uint8), _reference (contracts/transfers/DeBridgeGate.sol#1123) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

ReentrancyGuardUpgradable._gap (node_modules/openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradable.sol#68) is never used in DeBridgeGate (contracts/transfers/DeBridgeGate.sol#126-1145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DeBridgeGate.lockedClaim (contracts/DeBridgeGate.sol#102) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

grantRole(bytes32,address) should be declared external:
- AccessControlUpgradeable.grantRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#139-141)
revokeRole(bytes32,address) should be declared external:
- AccessControlUpgradeable.revokeRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#152-154)
renounceRole(bytes32,address) should be declared external:
- AccessControlUpgradeable.renounceRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#170-174)
name() should be declared external:
- ERC20.name() (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#162-64)
symbol() should be declared external:
- ERC20.symbol() (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#70-72)
decimals() should be declared external:
- ERC20.decimals() (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#87-89)
totalSupply() should be declared external:
- ERC20.totalSupply() (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#94-96)
balanceOf(address) should be declared external:
- ERC20.balanceOf(address) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#101-103)
transfer(address,uint256) should be declared external:
- ERC20.transfer(address,uint256) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#113-116)
allowance(address,address) should be declared external:
- ERC20.allowance(address,address) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#121-123)
approve(address,uint256) should be declared external:
- ERC20.approve(address,uint256) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#132-135)
transferFrom(address,address,uint256) should be declared external:
- ERC20.transferFrom(address,address,uint256) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#150-164)
increaseAllowance(address,uint256) should be declared external:
- ERC20.increaseAllowance(address,uint256) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#178-181)
decreaseAllowance(address,uint256) should be declared external:
- ERC20.decreaseAllowance(address,uint256) (node_modules/openzeppelin/contracts/token/ERC20/ERC20.sol#197-205)
initialize(uint8,IEETH) should be declared external:
- DeBridgeGate.initialize(uint8,IEETH) (contracts/transfers/DeBridgeGate.sol#168-177)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

DeBridgeTokenDeployer.sol

AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#222) shadows:
- ERC165Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36)
- ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
PausedUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/security/pausable/PausableUpgradeable.sol#97) shadows:
- ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
ERC20Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#362) shadows:
- ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) shadows:
- ERC20Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#97)
- ERC20Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#98)
- ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

ERC1967Upgrade._upgradeToAndCall(address,bytes,bool) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#64-73) ignores return value by Address.functionDelegateCall(newImplementation,data) (node_,rads.sol#71)
ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#108-108) ignores return value by Address.functionDelegateCall(newImplementation,data) (node_,rads.sol#71)
ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#108-108) ignores return value by Address.functionDelegateCall(newImplementation,abi.e) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#98-101)
ERC1967Upgrade._upgradeToAndCall(address,bytes,bool) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#183-193) ignores return value by Address.functionDelegateCall(IBEacon(newBeacon).imp)s/proxy/ERC1967/ERC1967Upgrade.sol#191)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

DeBridgeTokenDeployer.initialize(address,address,address) (contracts/transfers/DeBridgeTokenDeployer.sol#68-78) should emit an event for:
- DeBridgeTokenDeployer._debridgeAddress (node_modules/openzeppelin/contracts/transfers/DeBridgeTokenDeployer.sol#164-167) should emit an event for:
- _debridgeAddress = _debridgeAddress (contracts/transfers/DeBridgeTokenDeployer.sol#166)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

DeBridgeTokenDeployer.initialize(address,address,address), tokenImplementation (contracts/transfers/DeBridgeTokenDeployer.sol#69) lacks a zero-check on :
- tokenImplementation._tokenImplementation (contracts/transfers/DeBridgeTokenDeployer.sol#73)
DeBridgeTokenDeployer.initialize(address,address,_debridgeTokenAdmin) (contracts/transfers/DeBridgeTokenDeployer.sol#70) lacks a zero-check on :
- _debridgeTokenAdmin = _debridgeTokenAdmin (contracts/transfers/DeBridgeTokenDeployer.sol#74)
DeBridgeTokenDeployer.initialize(address,address,_debridgeAddress) (contracts/transfers/DeBridgeTokenDeployer.sol#71) lacks a zero-check on :
- _debridgeAddress = _debridgeAddress (contracts/transfers/DeBridgeTokenDeployer.sol#75)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#80-108):
External call:
- Address.functionDelegateCall(newImplementation,data) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#49)
- Address.functionDelegateCall(newImplementation,abi.encodeWithSignature("upgradeTo(address)",oldImplementation)) (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#98-101)
Event _upgradeToNewImplementation (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#56):
- _upgradeToNewImplementation (node_modules/openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

DeBridgeToken.permit(address,uint256,uint256,uint8,bytes32) (contracts/periphery/DeBridgeToken.sol#110-142) uses timestamp for comparisons
Dangerous comparisons:
- require(bool,string){deadline >= block.timestamp,permit: EXPIRED} (contracts/periphery/DeBridgeToken.sol#119)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

Proxy._delegate(address) (node_modules/openzeppelin/contracts/proxy/proxy/Proxy.sol#22-45) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/proxy/proxy/Proxy.sol#23-44)
Address.verifyCallResult(address,bytes,bytes) (node_modules/openzeppelin/contracts/utils/Address.sol#27-37) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/utils/Address.sol#33-35)
Address.verifyCallResult(bool,bytes,string) (node_modules/openzeppelin/contracts/utils/Address.sol#196-216) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/utils/Address.sol#208-211)
StorageSlot.getBooleanSlot(bytes32) (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#52-56) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#53-55)
StorageSlot.getBooleanSlot(bytes32) (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#61-65) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#66-68)
StorageSlot.getBooleanSlot(bytes32) (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#70-74) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#71-73)
StorageSlot.getUInt256Slot(bytes32) (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#79-83) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#84-88)
DeBridgeTokenDeployer._deployAsset(bytes32,string,string,uint8) (contracts/periphery/DeBridgeToken.sol#54-90) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#89-92)
DeBridgeTokenDeployer._deployAsset(bytes32,string,string,uint8) (contracts/transfers/DeBridgeTokenDeployer.sol#122-138) uses assembly
- INLINE ASM (node_modules/openzeppelin/contracts/storage/StorageSlot.sol#123-126)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

```



```

DeBridgeTokenDeployer.deployAsset(bytes32,string,string,uint8) (contracts/transfers/DeBridgeTokenDeployer.sol#85-138) uses literals with too many digits:
  - bytecode = abi.encodePacked(type(DeBridgeTokenProxy).creationCode, constructorArgs) (contracts/transfers/DeBridgeTokenDeployer.sol#120)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

ERC20PausableUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) is never used in DeBridgeToken (contracts/periphery/DeBridgeToken.sol#10-16)
AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#123) is never used in DeBridgeTokenDeployer (contracts/transfers/DeBridgeTokenDeployer.sol#12-16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

grantRole(bytes32,address) should be declared external:
  - AccessControlUpgradeable.grantRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#139-141)
RevokeRole(bytes32,address) should be declared external:
  - AccessControlUpgradeable.revokeRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#152-154)
renounceRole(bytes32,address) should be declared external:
  - AccessControlUpgradeable.renounceRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#176-178)
name() should be declared external:
  - ERC20Upgradeable.name() (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#68-70)
symbol() should be declared external:
  - ERC20Upgradeable.symbol() (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#76-78)
decimals() should be declared external:
  - DeBridgeToken.decimals() (contracts/periphery/DeBridgeToken.sol#145-147)
  - ERC20Upgradeable.decimals() (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#93-95)
totalSupply() should be declared external:
  - ERC20Upgradeable.totalSupply() (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#100-102)
balanceOf(address) should be declared external:
  - ERC20Upgradeable.balanceOf(address) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#107-109)
transfer(address,uint256) should be declared external:
  - ERC20Upgradeable.transfer(address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#119-122)
allowance(address,address) should be declared external:
  - ERC20Upgradeable.allowance(address,address) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#127-129)
approveFrom(address,address,uint256) should be declared external:
  - ERC20Upgradeable.approveFrom(address,address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#138-141)
transferFrom(address,address,uint256) should be declared external:
  - ERC20Upgradeable.transferFrom(address,address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#156-170)
increaseAllowance(address,uint256) should be declared external:
  - ERC20Upgradeable.increaseAllowance(address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#184-187)
decreaseAllowance(address,uint256) should be declared external:
  - ERC20Upgradeable.decreaseAllowance(address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#203-211)
pause() should be declared external:
  - DeBridgeToken.pause() (contracts/periphery/DeBridgeToken.sol#150-152)
unpause() should be declared external:
  - DeBridgeToken.unpause() (contracts/periphery/DeBridgeToken.sol#155-157)
initialize(address,address,address) should be declared external:
  - DeBridgeTokenDeployer.initialize(address,address,address) (contracts/transfers/DeBridgeTokenDeployer.sol#68-78)
implementor() should be declared external:
  - DeBridgeTokenDeployer.implementor() (contracts/transfers/DeBridgeTokenDeployer.sol#141-143)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

```

## OraclesManager.sol

```

AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#222) shadows:
  - ERC165Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36)
  - ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#51)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

Different versions of Solidity is used:
  - Version used: ('0.8.7', '+0.8.8')
    * 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4)
    * 0.8.1 (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#14)
    * 0.8.2 (node_modules/openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
    * 0.8.3 (node_modules/openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
    * 0.8.4 (node_modules/openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
    * 0.8.5 (node_modules/openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
    * 0.8.6 (node_modules/openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
    * 0.8.7 (node_modules/openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
    * 0.8.7 (contracts/interfaces/IOraclesManager.sol#2)
    * 0.8.7 (contracts/transfers/OraclesManager.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragmas-are-used

OraclesManager.addDracles(address[],bool[]) (contracts/transfers/OraclesManager.sol#78-101) has costly operations inside a loop:
  - requiredDraclesCount == 1 (contracts/transfers/OraclesManager.sol#92)
OraclesManager.updateDracle(address,bool) (contracts/transfers/OraclesManager.sol#107-139) has costly operations inside a loop:
  - updateDracle (contracts/transfers/OraclesManager.sol#107-139)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

AccessControlUpgradeable._AccessControl_init() (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#51-65) is never used and should be removed
AccessControlUpgradeable._AccessControl_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-68) is never used and should be removed
AccessControlUpgradeable._setRoleAdmin(bytes32,bytes32) (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#203-207) is never used and should be removed
AccessControlUpgradeable._setupRole(bytes32,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#194-196) is never used and should be removed
ContextUpgradeable._Context_init() (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#18-20) is never used and should be removed
ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#18-20) is never used and should be removed
ContextUpgradeable._msgData() (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#26-28) is never used and should be removed
ERC165Upgradeable._ERC165_init() (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#26-28) is never used and should be removed
ERC165Upgradeable._ERC165_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#26-28) is never used and should be removed
OraclesManager._addDracles(address[],bool[]) (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#128-130) is never used and should be removed
StringUpgradeable.toHexString(uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#48-51) is never used and should be removed
StringUpgradeable.toHexString(uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#135-35) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4) allows old versions
Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/IAccessControlUpgradeable.sol#4) allows old versions
Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4) allows old versions
Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/proxy/utils/UUPSUpgradeable.sol#4) allows old versions
Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#4) allows old versions
Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#4) allows old versions
Pragma version 0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/IERC165Upgradeable.sol#4) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Function AccessControlUpgradeable._AccessControl_init() (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#51-65) is not in mixedCase
Function AccessControlUpgradeable._AccessControl_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-68) is not in mixedCase
Variable AccessControlUpgradeable._roles (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#14) is not in mixedCase
Function ContextUpgradeable._Context_init() (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable._msgData (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#21) is not in mixedCase
Function ERC165Upgradeable._ERC165_init() (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable._ERC165_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#28-29) is not in mixedCase
Variable ERC165Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36) is not in mixedCase
Parameter OracleManager._initialize(uint8,uint8)_minConfirmation (contract/transfers/OraclesManager.sol#52) is not in mixedCase
Parameter OracleManager._initialize(uint8,uint8)_maxConfirmation (contract/transfers/OraclesManager.sol#53) is not in mixedCase
Parameter OracleManager._minDracles (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#41) is not in mixedCase
Parameter OracleManager._maxDracles (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#42) is not in mixedCase
Parameter OracleManager._excessConfirms (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#70) is not in mixedCase
Parameter OracleManager._addDracles(address[],bool)_oracles (contracts/transfers/OracleManager.sol#79) is not in mixedCase
Parameter OracleManager._addDracles(address[],bool)_draclesRequired (contracts/transfers/OracleManager.sol#104) is not in mixedCase
Parameter OracleManager._updateDracle(address,bool)_isValid (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#105) is not in mixedCase
Parameter OracleManager._updateDracle(address,bool)_dracleRequired (contracts/transfers/OracleManager.sol#109) is not in mixedCase
Parameter OracleManager._updateDracle(address,bool)_dracleRequired (contracts/transfers/OracleManager.sol#110) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#222) is never used in OraclesManager (contracts/transfers/OraclesManager.sol#10-148)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

```

**SignatureVerifier.sol**

```

AccessControlUpgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) shadowed:
  - ERC165Upgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/utils/Introspection/ERC165Upgradeable.sol#26)
  - ContextUpgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#51)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

SignatureVerifier.submit(bytes32,bytes,uint8) (contracts/transfers/SignatureVerifier.sol#59-114) uses a dangerous strict equality:
  - currentBlock == uint40(block.number) (contracts/transfers/SignatureVerifier.sol#100)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

SignatureVerifier.submit(bytes32,bytes,uint8).confirmations (contracts/transfers/SignatureVerifier.sol#70) is a local variable never initialized
SignatureVerifier.submit(bytes32,bytes,uint8).currentRequiredOracleCount (contracts/transfers/SignatureVerifier.sol#68) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#initialization-local-variables

SignatureVerifier.initialize(uint8,uint8,uint8,address) (contracts/transfers/SignatureVerifier.sol#45-54) should emit an event for:
  - _debridgeAddress == _debridgeAddress (contracts/transfers/SignatureVerifier.sol#53)
SignatureVerifier.setDebridgeAddress(address) (contracts/transfers/SignatureVerifier.sol#127-129) should emit an event for:
  - _debridgeAddress == _debridgeAddress (contracts/transfers/SignatureVerifier.sol#128)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

SignatureVerifier.initialize(uint8,uint8,uint8,address),_debridgeAddress (contracts/transfers/SignatureVerifier.sol#49) lacks a zero-check on:
  - _debridgeAddress == _debridgeAddress (contracts/transfers/SignatureVerifier.sol#53)
SignatureVerifier.setDebridgeAddress(address) (contracts/transfers/SignatureVerifier.sol#127) lacks a zero-check on:
  - _debridgeAddress == _debridgeAddress (contracts/transfers/SignatureVerifier.sol#128)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

SignatureUtil.parseSignature(bytes,uint256) (contract/libraries/SignatureUtil.sol#32-49) uses assembly
  - INLINE ASM (contracts/libraries/SignatureUtil.sol#41-45)
SignatureUtil.toJub256(bytes,uint256) (contract/libraries/SignatureUtil.sol#51-61) uses assembly
  - INLINE ASM (contracts/libraries/SignatureUtil.sol#88-90)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different version of Solidity is used:
  - Version used: 0.8.0 (Solidity 0.8.0)
  - Version used: 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#1)
  * 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#4)
  * 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4)
  * 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#1)
  * 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#4)
  * 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/IERC165Upgradeable.sol#4)
  * 0.8.0 (contracts/interfaces/OraclesManager.sol#2)
  * 0.8.7 (contracts/libraries/SignatureUtil.sol#12)
  * 0.8.7 (contracts/libraries/SignatureUtil.sol#2)
  - 0.8.7 (contracts/transfers/OraclesManager.sol#2)
  - 0.8.7 (contracts/transfers/SignatureVerifier.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

OraclesManager.addOracle(address,bool) (contracts/transfers/OraclesManager.sol#78-101) has costly operations inside a loop:
  - requiredOraclesCount += 1 (contracts/transfers/OraclesManager.sol#92)
OraclesManager.getOracleCount() (contracts/transfers/OraclesManager.sol#107-109) has costly operations inside a loop:
  - oracleAddresses.length (contracts/transfers/OraclesManager.sol#129)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

AccessControlUpgradeable (AccessControlInit) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#52-55) is never used and should be removed
AccessControlUpgradeable._AccessControl_init_unchained() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#57-58) is never used and should be removed
AccessControlUpgradeable._setRoleAdmin(bytes32,uint32) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#203-207) is never used and should be removed
ContextUpgradeable._Context_init() (node_modules/Openzeppelin/contracts-upgradeable/introspection/ContextUpgradeable.sol#1) is never used and should be removed
ContextUpgradeable._Context_init_unchained() (node_modules/Openzeppelin/contracts-upgradeable/introspection/ContextUpgradeable.sol#23) is never used and should be removed
ContextUpgradeable._msgData (node_modules/Openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#28-30) is never used and should be removed
ERC165Upgradeable._ERC165_init() (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#26-29) is never used and should be removed
ERC165Upgradeable._ERC165_init_unchained() (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#30-33) is never used and should be removed
StringUpgradeable.toHexString(uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#4-51) is never used and should be removed
StringUpgradeable.toString(uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#15-35) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Prisma version 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#4) allows old versions
Prisma version 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/IAccessControlUpgradeable.sol#4) allows old versions
Prisma version 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4) allows old versions
Prisma version 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#1) allows old versions
Prisma version 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/IERC165Upgradeable.sol#4) allows old versions
Prisma version 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/IERC165Upgradeable.sol#4) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Function AccessControlUpgradeable._AccessControl_init() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#51-55) is not in mixedCase
Function AccessControlUpgradeable._AccessControl_init_unchained() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#57-58) is not in mixedCase
Variable AccessControlUpgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#1) is not in mixedCase
Function AccessControlUpgradeable._setRoleAdmin(bytes32,uint32) (node_modules/Openzeppelin/contracts-upgradeable/introspection/AccessControlUpgradable.sol#203-207) is not in mixedCase
Function ContextUpgradeable._Context_init() (node_modules/Openzeppelin/contracts-upgradeable/introspection/ContextUpgradeable.sol#1) is not in mixedCase
Variable ContextUpgradeable._Context_init_unchained() (node_modules/Openzeppelin/contracts-upgradeable/introspection/ContextUpgradeable.sol#23) is not in mixedCase
Variable ContextUpgradeable._msgData (node_modules/Openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#28-30) is not in mixedCase
Variable ERC165Upgradeable._ERC165_init() (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#26-29) is not in mixedCase
Function ERC165Upgradeable._ERC165_init_unchained() (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#30-33) is not in mixedCase
Variable ERC165Upgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#26-29) is not in mixedCase
Variable ERC165Upgradeable._msgData (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#34) is not in mixedCase
Parameter SignatureUtil.getUnsignedMsg(bytes32), submissionId (contracts/libraries/SignatureUtil.sol#13) is not in mixedCase
Parameter SignatureUtil.splitSignature(bytes,signature) (contracts/libraries/SignatureUtil.sol#14) is not in mixedCase
Parameter SignatureUtil.toJub256(bytes,uint256) (contracts/libraries/SignatureUtil.sol#15) is not in mixedCase
Parameter SignatureUtil.toJub256(bytes,uint256)_bytes (contracts/libraries/SignatureUtil.sol#15) is not in mixedCase
Parameter SignatureUtil.toJub256(bytes,uint256)_offset (contracts/libraries/SignatureUtil.sol#15) is not in mixedCase
Parameter OraclesManager.initialize(uint8,uint8) _minConfirmation (contracts/transfers/OraclesManager.sol#62) is not in mixedCase
Parameter OraclesManager.initialize(uint8,uint8) _maxConfirmation (contracts/transfers/OraclesManager.sol#63) is not in mixedCase
Parameter OraclesManager.setExcessConfirmations(uint8) _airConfirmations (contracts/transfers/OraclesManager.sol#63) is not in mixedCase
Parameter OraclesManager.setExcessConfirmations(uint8) _excessConfirmations (contracts/transfers/OraclesManager.sol#70) is not in mixedCase
Parameter OraclesManager.addOracles(address),_oracles (contracts/transfers/OraclesManager.sol#79) is not in mixedCase
Parameter OraclesManager.setMinConfirms(uint8) _minConfirms (contracts/transfers/OraclesManager.sol#80) is not in mixedCase
Parameter OraclesManager.setMaxConfirms(uint8) _maxConfirms (contracts/transfers/OraclesManager.sol#81) is not in mixedCase
Parameter OraclesManager.updateOracleAddress(bool),_isValid (contracts/transfers/OraclesManager.sol#109) is not in mixedCase
Parameter OracleManager.updateOracleAddress(bool),_isValid (contracts/transfers/OraclesManager.sol#109) is not in mixedCase
Parameter OracleManager.updateOracleAddress(bool),_require (contracts/transfers/OraclesManager.sol#110) is not in mixedCase
Parameter SignatureVerifier._checkSignature(bytes,address,uint8,uint8,uint8) _threshold (contracts/transfers/SignatureVerifier.sol#47) is not in mixedCase
Parameter SignatureVerifier._initialize(uint8,uint8,uint8,address),_minConfirmation (contracts/transfers/SignatureVerifier.sol#47) is not in mixedCase
Parameter SignatureVerifier._initialize(uint8,uint8,uint8,address),_maxConfirmation (contracts/transfers/SignatureVerifier.sol#48) is not in mixedCase
Parameter SignatureVerifier._initialize(uint8,uint8,uint8,address),_debridgeAddress (contracts/transfers/SignatureVerifier.sol#49) is not in mixedCase
Parameter SignatureVerifier._initialize(uint8,uint8,uint8,address),_debridgeAddress (contracts/transfers/SignatureVerifier.sol#49) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes,uint8),_excessConfirmations (contracts/transfers/SignatureVerifier.sol#50) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes,uint8),_excessConfirmations (contracts/transfers/SignatureVerifier.sol#51) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes,uint8),_debridgeAddress (contracts/transfers/SignatureVerifier.sol#52) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes,uint8),_debridgeAddress (contracts/transfers/SignatureVerifier.sol#52) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes,uint8),_excessConfirmations (contracts/transfers/SignatureVerifier.sol#53) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes,uint8),_excessConfirmations (contracts/transfers/SignatureVerifier.sol#54) is not in mixedCase
Parameter SignatureVerifier._submit(32,bytes),_threshold (contracts/transfers/SignatureVerifier.sol#55) is not in mixedCase
Parameter SignatureVerifier._threshold(uint8),_threshold (contracts/transfers/SignatureVerifier.sol#56) is not in mixedCase
Parameter SignatureVerifier._isValidSignature(bytes32,bytes),_signature (contracts/transfers/SignatureVerifier.sol#136) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

AccessControlUpgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControl.sol#232) is never used in SignatureVerifier (contracts/transfers/SignatureVerifier.sol#9-157)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

grantRole(bytes32,address) should be declared external:
  - AccessControlUpgradeable.grantRole(bytes32,address) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#139-141)
revokeRole(bytes32,address) should be declared external:
  - AccessControlUpgradeable.revokeRole(bytes32,address) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#152-154)
removeRole(bytes32,address) should be declared external:
  - AccessControlUpgradeable.removeRole(bytes32,address) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradable.sol#170-174)
initialize(uint8,uint8,uint8,address) should be declared external:
  - SignatureVerifier.initialize(uint8,uint8,uint8,address) (contracts/transfers/SignatureVerifier.sol#45-54)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

WethGate.sol
WethGate._safeTransferETH(address,int256) (contracts/transfers/WethGate.sol#37-40) sends eth to arbitrary user
  - Dangerous calls:
    - success = _to.call{value : value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

Reentrancy in WethGate.withdraw(address,uint256) (contracts/transfers/WethGate.sol#31-35):
  - External call to withdraw(wad) (contracts/transfers/WethGate.sol#32)
    - _safeTransferETH(_receiver,_wad) (contracts/transfers/WethGate.sol#33)
      - (success) = _to.call{value : value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
    - External call to withdraw(wad) (contracts/transfers/WethGate.sol#33)
      - _safeTransferETH(_receiver,_wad) (contracts/transfers/WethGate.sol#33)
        - (success) = _to.call{value : value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
        - (success) = _to.call{value : value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
        - Event emitted after the call();
      - Withdrawal(_receiver,_wad) (contracts/transfers/WethGate.sol#33)
        - (success) = _to.call{value : value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Low level call in WethGate._safeTransferETH(address,uint256) (contracts/transfers/WethGate.sol#37-40):
  - (success) = _to.call{value : value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter WethGate.withdraw(address,uint256)._receiver (contracts/transfers/WethGate.sol#31) is not in mixedCase
Parameter WethGate.withdraw(address,uint256)._wad (contracts/transfers/WethGate.sol#31) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

```

## SimpleFeeProxy.sol

```

SimpleFeeProxy._safeTransferETH(address,uint256) (contracts/periphery/SimpleFeeProxy.sol#121-124) sends eth to arbitrary user
  Dangerous calls:
    - (success) = to.call.value: value(uint bytes9) (contracts/periphery/SimpleFeeProxy.sol#122)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#232) shadows:
  - ERC165Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36)
  - ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
PausableUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
  - ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
  - ContextUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

SimpleFeeProxy.initialize(UDerivedData,address) (contracts/periphery/SimpleFeeProxy.sol#36) lacks a zero-check on :
  - treasury = treasury (contracts/periphery/SimpleFeeProxy.sol#88)
SimpleFeeProxy.setTreasury(address) (contracts/periphery/SimpleFeeProxy.sol#65) lacks a zero-check on :
  - treasury = treasury (contracts/periphery/SimpleFeeProxy.sol#67)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

AddressUpgradeable.isContract(address) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#27-37) uses assembly
  - INLINE_ASM (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#3-35)
AddressUpgradeable.isTypeCalleeResult(bool,bytes,bytes) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189) uses assembly
  - INLINE_ASM (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189)
SimpleFeeProxy.getChainId() (contracts/periphery/SimpleFeeProxy.sol#110-114) uses assembly
  - INLINE_ASM (contracts/interfaces/IDeBridgeGate.sol#11-13)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different version of Solidity is used:
  Version used: ('0.8.7', '+0.8.0')
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#6)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#14)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#4)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#4)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#4)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#4)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#4)
  *+0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#4)
  *+0.8.7 (contracts/interfaces/IDeBridgeGate.sol#2)
  *+0.8.7 (contracts/interfaces/IDeBridgeGate.sol#2)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

AccessControlUpgradeable._AccessControl_init() (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#51-55) is never used and should be removed
AccessControlUpgradeable._AccessControl_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#57-58) is never used and should be removed
AccessControlUpgradeable._Context_init() (node_modules/openzeppelin/contracts-upgradeable/context/ContextUpgradeable.sol#2-3) is never used and should be removed
AddressUpgradeable.functionCallWithValue(address,bytes,uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#109-115) is never used and should be removed
AddressUpgradeable.functionStaticCallWithValue(address,bytes) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#142-144) is never used and should be removed
AddressUpgradeable.functionStaticCallWith.selector(address,bytes) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#145-146) is never used and should be removed
AddressUpgradeable.functionStaticCallWith.selector(address,bytes,uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#155-156) is never used and should be removed
AddressUpgradeable.functionStaticCallWith.selector(address,bytes,uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#157-158) is never used and should be removed
ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is never used and should be removed
ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is never used and should be removed
ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#28-29) is never used and should be removed
ERC165Upgradeable._ERC165_init() (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#28-31) is never used and should be removed
ERC165Upgradeable._ERC165_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
PausableUpgradeable._Pauseable_init() (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#37) is never used and should be removed
SafeERC20Upgradeable.safeApprove(IERC20Upgradeable,address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/AddressUpgradeable.sol#45-48) is never used and should be removed
SafeERC20Upgradeable.safeDecreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/AddressUpgradeable.sol#65-68) is never used and should be removed
SafeERC20Upgradeable.safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/AddressUpgradeable.sol#66-67) is never used and should be removed
SafeERC20Upgradeable.safeTransfer(IERC20Upgradeable,address,uint256) (node_modules/openzeppelin/contracts-upgradeable/token/ERC20/AddressUpgradeable.sol#25-28) is never used and should be removed
StringUpgradeable.toString(uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#15-35) is never used and should be removed
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version<=0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#4) allows old versions
Pragma version>0.8.0 (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#4) allows old versions
Pragma version<=0.8.0 (node_modules/openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4) allows old versions
Pragma version>0.8.0 (node_modules/openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4) allows old versions
Pragma version<=0.8.0 (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#4) allows old versions
Pragma version>0.8.0 (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#4) allows old versions
Pragma version<=0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#4) allows old versions
Pragma version>0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#4) allows old versions
Pragma version<=0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#4) allows old versions
Pragma version>0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#4) allows old versions
Pragma version<=0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#4) allows old versions
Pragma version>0.8.0 (node_modules/openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#4) allows old versions
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Low level call in AddressUpgradeable.sendValue(address,uint256) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#55-60):
  - (success) = recipient.call.value: amount() (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#56)
Low level call in AddressUpgradeable.functionCallWithValue(address,bytes,uint256,string) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#123-134):
  - (success) = target.functionCallWithValue(address,bytes,uint256,string) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
Low level call in AddressUpgradeable.functionStaticCallWithValue(address,bytes,string) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161):
  - (success,returnData) = target.staticCallWithValue(address,bytes,string) (node_modules/openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#159)
Low level call in SimpleFeeProxy._safeTransferETH(address,uint256) (contracts/periphery/SimpleFeeProxy.sol#121-124):
  - (success) = recipient.call.value: value(uint bytes9) (contracts/periphery/SimpleFeeProxy.sol#122)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Function AccessControlUpgradeable._AccessControl_init() (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#51-55) is not in mixedCase
Function AccessControlUpgradeable._AccessControl_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#57-58) is not in mixedCase
Variable AccessControlUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable._Pauseable_init() (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable._Pauseable_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable ContextUpgradeable._Context_init() (node_modules/openzeppelin/contracts-upgradeable/context/ContextUpgradeable.sol#2-3) is not in mixedCase
Function ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/context/ContextUpgradeable.sol#22-23) is not in mixedCase
Function ContextUpgradeable._Context_init_unchained() (node_modules/openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ERC165Upgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/introspection/ERC165Upgradeable.sol#28-29) is not in mixedCase
Function SimpleFeeProxy.initialize(IDeBridgeGate,address) (contracts/periphery/SimpleFeeProxy.sol#6) is not in mixedCase
Parameter SimpleFeeProxy.initialize(IDeBridgeGate,address) (contracts/periphery/SimpleFeeProxy.sol#6) is not in mixedCase
Parameter SimpleFeeProxy.initialize(IDeBridgeGate,address) (contracts/periphery/SimpleFeeProxy.sol#121-124) is not in mixedCase
Parameter SimpleFeeProxy.setTreasury(address) (contracts/periphery/SimpleFeeProxy.sol#65) is not in mixedCase
Parameter SimpleFeeProxy.withdrawFee(address) (contracts/periphery/SimpleFeeProxy.sol#62) is not in mixedCase
Parameter SimpleFeeProxy.withdrawFee(address) (contracts/periphery/SimpleFeeProxy.sol#121-124) is not in mixedCase
Parameter SimpleFeeProxy._tokenAddress (contracts/periphery/SimpleFeeProxy.sol#94) is not in mixedCase
Parameter SimpleFeeProxy._chainId (contracts/periphery/SimpleFeeProxy.sol#105) is not in mixedCase
Parameter SimpleFeeProxy._tokenAddress (contracts/periphery/SimpleFeeProxy.sol#105) is not in mixedCase
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

PausableUpgradeable._gap (node_modules/openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is never used in SimpleFeeProxy (contracts/periphery/SimpleFeeProxy.sol#12-131)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

grantRole(bytes2,address) should be declared external:
  - AccessControlUpgradeable.grantRole(bytes2,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#139-141)
revokeRole(bytes2,address) should be declared external:
  - AccessControlUpgradeable.revokeRole(bytes2,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#152-154)
renounceRole(bytes2,address) should be declared external:
  - AccessControlUpgradeable.renounceRole(bytes2,address) (node_modules/openzeppelin/contracts-upgradeable/access/AccesControlUpgradeable.sol#176-177)
initialize(IDeBridgeGate,address) should be declared external:
  - SimpleFeeProxy.initialize(IDeBridgeGate,address) (contracts/periphery/SimpleFeeProxy.sol#36-48)
  Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

```

## DefiController.sol

```

AccessControlUpgradeable_gpp (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) shadows:
- ERC165Upgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#36)
- ContextUpgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36)
PausableUpgradeable_gpp (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
- ContextUpgradeable._gap (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ContextUpgradeable.sol#31)
Reentrancy in DefiController.depositToStrategy(uint256,address) (contracts/periphery/DefiController.sol#91-117):
- dBridgeGate.requestReserves(strategy.stakeToken_,amount) (contracts/periphery/DefiController.sol#109)
- IERC20Upgradeable(strategy.stakeToken_,safeApprove(address,integrateStrategyController,_amount)) (contracts/periphery/DefiController.sol#112)
- strategyController.depositToStrategy(stakeToken_,amount) (contracts/periphery/DefiController.sol#124)
Event emitted after the call:
- DepositToStrategy(_strategy,_amount) (contracts/periphery/DefiController.sol#116)
Reentrancy in DefiController.withdrawFromStrategy(uint256,address) (contracts/periphery/DefiController.sol#119-136):
- External call:
  - strategyController.withdrawStrategy(strategyToken_,amount) (contracts/periphery/DefiController.sol#126)
  - IERC20Upgradeable(strategy.stakeToken_,safeApprove(address,integrateBridgeGate,_amount)) (contracts/periphery/DefiController.sol#127)
  - IERC20Upgradeable(strategy.stakeToken_,safeApprove(address,integrateBridgeGate,_amount)) (contracts/periphery/DefiController.sol#128)
- strategyController.withdrawFromStrategy(stakeToken_,amount) (contracts/periphery/DefiController.sol#129)
Event emitted after the call:
- WithdrawFromStrategy(_strategy,_amount) (contracts/periphery/DefiController.sol#135)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#state-variable-shadowing

Reentrancy in DefiController.depositToStrategy(uint256,address) (contracts/periphery/DefiController.sol#91-117):
- dBridgeGate.requestReserves(strategy.stakeToken_,amount) (contracts/periphery/DefiController.sol#109)
- IERC20Upgradeable(strategy.stakeToken_,safeApprove(address,integrateStrategyController,_amount)) (contracts/periphery/DefiController.sol#112)
- strategyController.depositToStrategy(stakeToken_,amount) (contracts/periphery/DefiController.sol#124)
Event emitted after the call:
- DepositToStrategy(_strategy,_amount) (contracts/periphery/DefiController.sol#116)
Reentrancy in DefiController.withdrawFromStrategy(uint256,address) (contracts/periphery/DefiController.sol#119-136):
- External call:
  - strategyController.withdrawStrategy(strategyToken_,amount) (contracts/periphery/DefiController.sol#126)
  - IERC20Upgradeable(strategy.stakeToken_,safeApprove(address,integrateBridgeGate,_amount)) (contracts/periphery/DefiController.sol#127)
  - IERC20Upgradeable(strategy.stakeToken_,safeApprove(address,integrateBridgeGate,_amount)) (contracts/periphery/DefiController.sol#128)
- strategyController.withdrawFromStrategy(stakeToken_,amount) (contracts/periphery/DefiController.sol#129)
Event emitted after the call:
- WithdrawFromStrategy(_strategy,_amount) (contracts/periphery/DefiController.sol#135)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

AddressUpgradeable_isContract(address) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#27-37) uses assembly
- INLINE ASM (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#33-35)
AddressUpgradeable_isNotContract(address) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189) uses assembly
- INLINE ASM (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#181-184)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#assembly

Different versions of Solidity is used:
- Version used: '0.8.7' '^0.8.0'
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#4)
- 0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#4)
- 0.8.7 (contracts/interfaces/IStrategy.sol#2)
- 0.8.7 (contracts/interfaces/IUniswapV2Factory.sol#2)
- 0.8.7 (contracts/interfaces/IUniswapV2Pair.sol#2)
- 0.8.7 (contracts/interfaces/IUniswapV2Router01.sol#2)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#different-pragma-directives-are-used

AccessControlUpgradeable__AccessControl_init() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#51-55) is never used and should be removed
AccessControlUpgradeable__AccessControl_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-58) is never used and should be removed
AccessControlUpgradeable_setRoleAdmin(bytes32,bytes32) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#203-207) is never used and should be removed
AddressUpgradeable_functionCallWithValue(address,bytes,uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#80-82) is never used and should be removed
AddressUpgradeable_functionCallWithValue(address,bytes,uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#189-195) is never used and should be removed
AddressUpgradeable_functionStaticCallWithValue(string,bytes) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-151) is never used and should be removed
AddressUpgradeable_sendValue(address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#55-60) is never used and should be removed
ContextUpgradeable__Context_init() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#18-20) is never used and should be removed
ContextUpgradeable_Context_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#21-22) is never used and should be removed
ContextUpgradeable_Context_init_.uninitialized() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#23-30) is never used and should be removed
ContextUpgradeable_Context_init_.uninitialized() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#24-26) is never used and should be removed
ERC165Upgradeable__ERC165_init() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
ERC165Upgradeable__ERC165_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
PausableUpgradeable_Pausable_init() (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#41-42) is never used and should be removed
PausableUpgradeable_Pausable_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#41-42) is never used and should be removed
SafeERC20Upgradeable_safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#69-70) is never used and should be removed
SafeERC20Upgradeable_safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#69-67) is never used and should be removed
SafeERC20Upgradeable_safeTransfer(IERC20Upgradeable,address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#121-122) is never used and should be removed
SafeERC20Upgradeable_safeTransfer(IERC20Upgradeable,address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#121-127) is never used and should be removed
StringUpgradeable_toString(uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#40-51) is never used and should be removed
StringUpgradeable_toString(uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/StringUpgradeable.sol#55-56) is never used and should be removed
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#dead-code

Pragma version=0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4) allows old versions
Pragma version=0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/access/IAccessControlUpgradeable.sol#4) allows old versions
Pragma version=0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4) allows old versions
Pragma version=0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/proxy/Initializable.sol#4) allows old versions
Pragma version=0.8.0 (node_modules/Openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#4) allows old versions
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#incorrect-versions-in-solidity

Low level call in AddressUpgradeable_sendValue(address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#55-60):
- (success).recipient.call.value(amount) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#56)
Low level call in AddressUpgradeable_sendValue(address,uint256) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#123-134):
- (success,returnData) = target.call.value(value)(data) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#123)
Low level call in AddressUpgradeable_functionStaticCallWithValue(string,bytes) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161):
- (success,returnData) = target.staticcall(data) (node_modules/Openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#159)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#low-level-calls

Function AccessControlUpgradeable__AccessControl_init() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#51-55) is not in mixedCase
Function AccessControlUpgradeable__AccessControl_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-58) is not in mixedCase
Variable PausableUpgradeable_Pausable_init() (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#41-42) is not in mixedCase
Function PausableUpgradeable_Pausable_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#41-41) is not in mixedCase
Variable PausableUpgradeable__Pausable_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#41-41) is not in mixedCase
Function ContextUpgradeable_Context_init() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable_Context_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#21-22) is not in mixedCase
Variable ContextUpgradeable__Context_init_.uninitialized() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ContextUpgradeable.sol#23-30) is not in mixedCase
Function ERC165Upgradeable__ERC165_init() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable__ERC165_init_.unchained() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ERC165Upgradeable.sol#28-29) is not in mixedCase
Variable ERC165Upgradeable__ERC165_init_.uninitialized() (node_modules/Openzeppelin/contracts-upgradeable/utility/introspection/ERC165Upgradeable.sol#28-29) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/interfaces/IUniswapV2Pair.sol#80) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/interfaces/IUniswapV2Pair.sol#82) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/interfaces/IUniswapV2Pair.sol#83) is not in mixedCase
Parameter DefiController.depositToStrategy(uint256,address),_strategy (contracts/periphery/DefiController.sol#91) is not in mixedCase
Parameter DefiController.withdrawFromStrategy(uint256,address),_amount (contracts/periphery/DefiController.sol#119) is not in mixedCase
Parameter DefiController.withdrawFromStrategy(uint256,address),_amount (contracts/periphery/DefiController.sol#120) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address),_strategy (contracts/periphery/DefiController.sol#124) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address),_strategy (contracts/periphery/DefiController.sol#125) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address),_stakeToken (contracts/periphery/DefiController.sol#207) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address),_stakeToken (contracts/periphery/DefiController.sol#209) is not in mixedCase
Parameter DefiController.updateStrategy(address,bool,uint16),_strategyToken (contracts/periphery/DefiController.sol#242) is not in mixedCase
Parameter DefiController.updateStrategy(address,bool,uint16),_strategyToken (contracts/periphery/DefiController.sol#243) is not in mixedCase
Parameter DefiController.updateStrategy(address,bool,uint16),_maxReservesBps (contracts/periphery/DefiController.sol#242) is not in mixedCase
Parameter DefiController.updateStrategy(address),_deBridgeGate (contracts/periphery/DefiController.sol#265) is not in mixedCase
Parameter DefiController.addWorker(address),_worker (contracts/periphery/DefiController.sol#268) is not in mixedCase
Parameter DefiController.addWorker(address),_worker (contracts/periphery/DefiController.sol#269) is not in mixedCase
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

PausableUpgradeable_gpp (node_modules/Openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is never used in DefiController (contracts/periphery/DefiController.sol#14-29)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#unused-state-variable

renounceRole(bytes32,address) should be declared external:
- AccessControlUpgradeable.renounceRole(bytes32,address) (node_modules/Openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#178-174)
initializersDefiController_initialize() (contracts/periphery/DefiController.sol#62-69)
- DefiController.initialize() (contracts/periphery/DefiController.sol#62-69)
Reference: https://github.com/crytic/lithier/wiki/Detector-Documentation#public-function-that-could-be-declared-external

```

## CallProxy.sol



## DeBridgeToken.sol







# AUTOMATED TESTING

### MultiSendCallOnly.sol

```
MultisendCallOnly._multiSend(bytes) (contracts/libraries/MultiSendCallOnly.sol#21-60) uses assembly
  - INLINE ASM (contracts/libraries/MultiSendCallOnly.sol#23-69)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

MultisendCallOnly._multiSend(bytes) (contracts/libraries/MultiSendCallOnly.sol#21-60) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version=>0.7.0<0.9.0 (contracts/libraries/MultiSendCallOnly.sol#2) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

- As a result of the tests carried out with the Slither tool, some results were obtained and reviewed by Halborn. Based on the results reviewed, some vulnerabilities were determined to be false positives. The actual vulnerabilities found by Slither are already included in the report findings.

## 4.2 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 all the contracts and sent the compiled results to the analyzers to locate any vulnerabilities.

### MythX results:

#### `SignatureVerifier.sol`

Line	SWC Title	Severity	Short Description
100	(SWC-128) Weak Sources of Randomness from Chain Attributes	Low	Potential use of "block.number" as source of randomness.
103	(SWC-128) Weak Sources of Randomness from Chain Attributes	Low	Potential use of "block.number" as source of randomness.

#### `WethGate.sol`

Line	SWC Title	Severity	Short Description
32	(SWC-107) Reentrancy	Low	A call to a user-supplied address is executed.
38	(SWC-118) DoS with Failed Call	Low	Multiple calls are executed in the same transaction.
38	(SWC-107) Reentrancy	Low	A call to a user-supplied address is executed.

#### `SimpleFeeProxy.sol`

Line	SWC Title	Severity	Short Description
12	(SWC-123) Requirement Violation	Low	Requirement violation.
65	(SWC-123) Requirement Violation	Low	Requirement violation.
82	(SWC-107) Reentrancy	Low	Read of persistent state following external call.
88	(SWC-107) Reentrancy	Low	Read of persistent state following external call.

#### `DeBridgeToken.sol`

Line	SWC Title	Severity	Short Description
119	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The <code>block.timestamp</code> environment variable.

#### `MultiSendCallOnly.sol`

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

- No major issues found by Mythx. The floating pragma flagged by MythX is a false positive, as the pragma is set in the `hardhat.config.ts` file to the `0.8.7` version.

THANK YOU FOR CHOOSING  
HALBORN