DeBridge - Solidity

Smart Contract Security Audit

Prepared by: **Halborn**

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

Visit: **Halborn.com**

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.

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
- SignatureUtil.sol
- BytesLib.sol

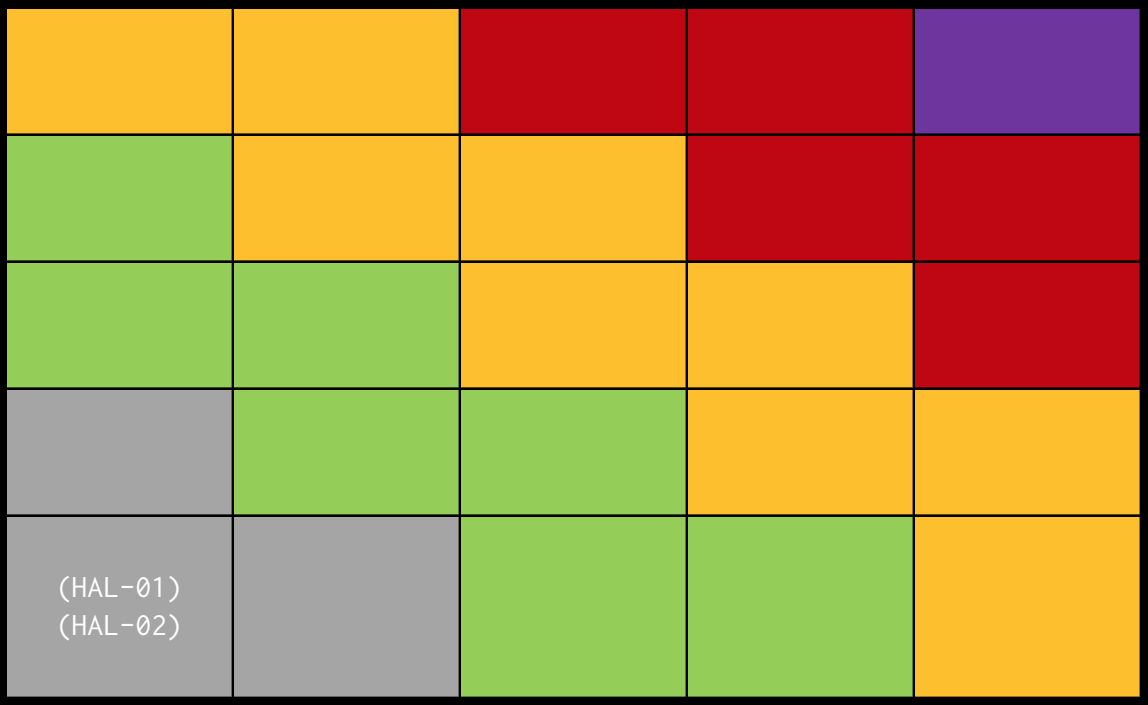
Commit ID: 47335f5f8f46ccd52594d594cb1d2c7a1d6dcf2c

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,
    ↳ _excessConfirmations);
52      confirmationThreshold = _confirmationThreshold;
53      debridgeAddress = _debridgeAddress;
54  }
```

Listing 11: SignatureVerifier.sol (Line 128)

```
127 function setDebridgeAddress(address _debridgeAddress) external
    ↳ 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
    ↳ ) 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 }
371
```

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
 ↳  WrongArgument();
394     DebridgeFeeInfo storage debridgeFee = getDebridgeFeeInfo[
 ↳ _debridgeId];
395     for (uint256 i = 0; i < _supportedChainIds.length; i++) {
396         debridgeFee.getChainFee[_supportedChainIds[i]] =
 ↳ _assetFeesInfo[i];
397     }
398 }
399
```

Listing 24: DeBridgeGate.sol (Line 555)

```
554 function blockSubmission(bytes32[] memory _submissionIds, bool
 ↳ isBlocked) external onlyAdmin {
555     for (uint256 i = 0; i < _submissionIds.length; i++) {
556         isBlockedSubmission[_submissionIds[i]] = isBlocked;
557         if (isBlocked) {
558             emit Blocked(_submissionIds[i]);
559         } else {
560             emit Unblocked(_submissionIds[i]);
561         }
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
 ↳ WrongArgument();
177     for (uint256 i = 0; i < _debridgeIds.length; i++) {
178         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.
 ↳ length) / 2 + 1) revert LowMinConfirmations();
 84
 85     for (uint256 i = 0; i < _oracles.length; i++) {
 86         OracleInfo storage oracleInfo = getOracleInfo[_oracles[i
 ↳ ]];
 87         if (oracleInfo.exist) revert OracleAlreadyExist();
 88
 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
 ↳ keeping an order
125         for (uint256 i = 0; i < oracleAddresses.length; i++) {
126             if (oracleAddresses[i] == _oracle) {
127                 oracleAddresses[i] = oracleAddresses[
 ↳ oracleAddresses.length - 1];
128                 oracleAddresses.pop();
129                 break;
130             }
131         }
132     } else if (!oracleInfo.isValid && _isValid) {
133         if (minConfirmations < (oracleAddresses.length + 1) / 2 +
 ↳ 1) revert LowMinConfirmations();
134         oracleAddresses.push(_oracle);
135     }
136     oracleInfo.isValid = _isValid;
137     oracleInfo.required = _required;
138     emit UpdateOracle(_oracle, _required, _isValid);
139 }
140
```

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 >
 ↳ minConfirmations
65         ? _excessConfirmations
66         : minConfirmations;
```

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

```
106         emit SubmissionApproved(_submissionId);
107     }
108
109     if (submissionsInBlock > confirmationThreshold) {
110         if (confirmations < excessConfirmations) revert
 ↳ NotConfirmedThreshold();
111     }
112
113     if (confirmations < needConfirmations) revert
 ↳ SubmissionNotConfirmed();
114 }
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
↳ chainId,address verifyingContract)"
83             ),
84             keccak256(bytes(name_)),
85             keccak256(bytes("1")),
86             chainId,
87             address(this)
88         )
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 }
86
```

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 }
106
```

Listing 32: Claimer.sol (Line 143)

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

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#659-804):
        External calls:
        - _validateToken(_tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,None) = _token.call(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#726)
        - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/transfers/DeBridgeGate.sol#731)
        - _safeTransferETH(msg.sender,msg.value - nativeFee) (contracts/transfers/DeBridgeGate.sol#769)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        External calls sending eth:
        - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
        - _safeTransferETH(msg.sender,msg.value - nativeFee) (contracts/transfers/DeBridgeGate.sol#769)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        State variables written after the call(s):
        - debridge.balance += amountAfterFee (contracts/transfers/DeBridgeGate.sol#797)
        - debridge.balance -= amountAfterFee (contracts/transfers/DeBridgeGate.sol#800)
        - getDebridgeFeeInfo[nativeDebridgeId].collectedFees += 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#31)
PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
        - ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

DeBridgeGate._normalizeTokenAmount(address,uint256) (contracts/transfers/DeBridgeGate.sol#1011-1024) performs a multiplication on the result of a division:
        -_amount = _amount / multiplier * 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,uint32,bytes) (contracts/transfers/DeBridgeGate.sol#182-229) 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.deployNewAsset(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)
        - deBridgeTokenAddress = IDeBridgeTokenDeployer(deBridgeTokenDeployer).deployAsset(debridgeId,_name,_symbol,_decimals) (contracts/transfers/DeBridgeGate.sol#334-335)
        State variables written after the call(s):
        - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
                - debridge.exist = true (contracts/transfers/DeBridgeGate.sol#628)
                - debridge.tokenAddress = _tokenAddress (contracts/transfers/DeBridgeGate.sol#629)
                - debridge.chainId = _nativeChainId (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-1

DeBridgeGate.claim(bytes32,uint256,uint256,address,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,uint32,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.setDefiController(address) (contracts/transfers/DeBridgeGate.sol#462-464) should emit an event for:
        - defiController = _defiController (contracts/transfers/DeBridgeGate.sol#463)
DeBridgeGate.setFeeContractUpdater(address) (contracts/transfers/DeBridgeGate.sol#468-470) should emit an event for:
        - feeContractUpdater = _value (contracts/transfers/DeBridgeGate.sol#469)
DeBridgeGate.setFeeProxy(address) (contracts/transfers/DeBridgeGate.sol#547-549) should emit an event for:
        - feeProxy = _feeProxy (contracts/transfers/DeBridgeGate.sol#548)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

DeBridgeGate.initialize(uint8,IWETH) (contracts/transfers/DeBridgeGate.sol#168-177) should emit an event for:
        - excessConfirmations = _excessConfirmations (contracts/transfers/DeBridgeGate.sol#172)
DeBridgeGate.updateExcessConfirmations(uint8) (contracts/transfers/DeBridgeGate.sol#402-405) should emit an event for:
        - excessConfirmations = _excessConfirmations (contracts/transfers/DeBridgeGate.sol#404)
DeBridgeGate.updateFlashFee(uint256) (contracts/transfers/DeBridgeGate.sol#567-570) 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 :
```

```
                - callProxy = _callProxy (contracts/transfers/DeBridgeGate.sol#424)
DeBridgeGate.setSignatureVerifier(address)._verifier (contracts/transfers/DeBridgeGate.sol#450) lacks a zero-check on :
                - signatureVerifier = _verifier (contracts/transfers/DeBridgeGate.sol#451)
DeBridgeGate.setDeBridgeTokenDeployer(address)._deBridgeTokenDeployer (contracts/transfers/DeBridgeGate.sol#456) lacks a zero-check on :
                - deBridgeTokenDeployer = _deBridgeTokenDeployer (contracts/transfers/DeBridgeGate.sol#457)
DeBridgeGate.setDefiController(address)._defiController (contracts/transfers/DeBridgeGate.sol#462) lacks a zero-check on :
                - defiController = _defiController (contracts/transfers/DeBridgeGate.sol#463)
DeBridgeGate.setFeeContractUpdater(address)._value (contracts/transfers/DeBridgeGate.sol#468) lacks a zero-check on :
                - feeContractUpdater = _value (contracts/transfers/DeBridgeGate.sol#469)
DeBridgeGate.setFeeProxy(address)._feeProxy (contracts/transfers/DeBridgeGate.sol#547) lacks a zero-check on :
                - feeProxy = _feeProxy (contracts/transfers/DeBridgeGate.sol#548)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in DeBridgeGate._send(bytes,address,uint256,uint256,bool) (contracts/transfers/DeBridgeGate.sol#659-804):
        External calls:
        - _validateToken(_tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,None) = _token.call(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)
        State variables written after the call(s):
        - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
                - getAmountThreshold[_debridgeId] = type()(uint256).max (contracts/transfers/DeBridgeGate.sol#638)
        - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
                - debridge.exist = true (contracts/transfers/DeBridgeGate.sol#628)
                - debridge.tokenAddress = _tokenAddress (contracts/transfers/DeBridgeGate.sol#629)
                - debridge.chainId = _nativeChainId (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)
        - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
                - tokenInfo.nativeChainId = _nativeChainId (contracts/transfers/DeBridgeGate.sol#642)
                - tokenInfo.nativeAddress = _nativeAddress (contracts/transfers/DeBridgeGate.sol#643)
Reentrancy in DeBridgeGate.deployNewAsset(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)
        - deBridgeTokenAddress = IDeBridgeTokenDeployer(deBridgeTokenDeployer).deployAsset(debridgeId,_name,_symbol,_decimals) (contracts/transfers/DeBridgeGate.sol#334-335)
        State variables written after the call(s):
        - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
                - getAmountThreshold[_debridgeId] = type()(uint256).max (contracts/transfers/DeBridgeGate.sol#638)
        - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
                - tokenInfo.nativeChainId = _nativeChainId (contracts/transfers/DeBridgeGate.sol#642)
                - tokenInfo.nativeAddress = _nativeAddress (contracts/transfers/DeBridgeGate.sol#643)
Reentrancy in DeBridgeGate.flash(address,address,uint256,bytes) (contracts/transfers/DeBridgeGate.sol#287-308):
        External calls:
        - IERC20Upgradeable(_tokenAddress).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#299)
        - IFlashCallback(msg.sender).flashCallback(currentFlashFee,_data) (contracts/transfers/DeBridgeGate.sol#300)
        State variables written after the call(s):
        - getDebridgeFeeInfo[debridgeId].collectedFees += paid (contracts/transfers/DeBridgeGate.sol#306)
Reentrancy in DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes) (contracts/transfers/DeBridgeGate.sol#182-229):
        External calls:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - IERC20Permit(_tokenAddress).permit(msg.sender,address(this),_amount,deadline,v,r,s) (contracts/transfers/DeBridgeGate.sol#677-684)
                - (success,None) = _token.call(abi.encodeWithSignature(symbol())) (contracts/transfers/DeBridgeGate.sol#889)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
                - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/transfers/DeBridgeGate.sol#731)
                - IDeBridgeToken(debridge.tokenAddress).burn(amountAfterFee) (contracts/transfers/DeBridgeGate.sol#801)
        External calls sending eth:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
        State variables written after the call(s):
        - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
                - nonce ++ (contracts/transfers/DeBridgeGate.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in DeBridgeGate._claim(bytes32,bytes32,address,uint256,uint256,IDeBridgeGate.SubmissionAutoParamsFrom) (contracts/transfers/DeBridgeGate.sol#898-966):
        External calls:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
                - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
                - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
        - status = ICallProxy(_callProxy).call(_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#931-938)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
                - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
        - status = ICallProxy(_callProxy).callERC20(_token,_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#943-9
        External calls sending eth:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        Event emitted after the call(s):
        - AutoRequestExecuted(_submissionId,status,_callProxy) (contracts/transfers/DeBridgeGate.sol#953)
Reentrancy in DeBridgeGate._claim(bytes32,bytes32,address,uint256,uint256,IDeBridgeGate.SubmissionAutoParamsFrom) (contracts/transfers/DeBridgeGate.sol#898-966):
        External calls:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
                - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
                - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
        - status = ICallProxy(_callProxy).call(_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#931-938)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
                - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - status = ICallProxy(_callProxy).callERC20(_token,_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#943-9
        - _withdrawWeth(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#956)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
                - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
        - _mintOrTransfer(_token,_receiver,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#958)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
                - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        External calls sending eth:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#956)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
```

```
        - _mintOrTransfer(_token,_receiver,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#958)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        Event emitted after the call(s):
        - MonitoringClaimEvent(_submissionId,debridge.balance,IERC20Upgradeable(debridge.tokenAddress).totalSupply()) (contracts/transfers/DeBridgeGate.sol#961-965)
Reentrancy in DeBridgeGate._send(bytes,address,uint256,uint256,bool) (contracts/transfers/DeBridgeGate.sol#659-804):
        External calls:
        - _validateToken(_tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,None) = _token.call(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)
        Event emitted after the call(s):
        - PairAdded(_debridgeId,_tokenAddress,_nativeAddress,_nativeChainId,debridge.maxAmount,debridge.minReservesBps) (contracts/transfers/DeBridgeGate.sol#645-652)
                - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
Reentrancy in DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes) (contracts/transfers/DeBridgeGate.sol#232-284):
        External calls:
        - _checkConfirmations(submissionId,_debridgeId,_amount,_signatures) (contracts/transfers/DeBridgeGate.sol#263)
                - ISignatureVerifier(signatureVerifier).submit(_submissionId,_signatures,excessConfirmations) (contracts/transfers/DeBridgeGate.sol#605-609)
                - ISignatureVerifier(signatureVerifier).submit(_submissionId,_signatures,0) (contracts/transfers/DeBridgeGate.sol#605-609)
        - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
                - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
                - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
                - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
                - status = ICallProxy(_callProxy).call(_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#931-938)
                - status = ICallProxy(_callProxy).callERC20(_token,_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.s
        External calls sending eth:
        - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        Event emitted after the call(s):
        - AutoRequestExecuted(_submissionId,status,_callProxy) (contracts/transfers/DeBridgeGate.sol#953)
                - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
        - Claimed(submissionId,_debridgeId,_amount,_receiver,_nonce,_chainIdFrom,_autoParams,isNativeToken) (contracts/transfers/DeBridgeGate.sol#274-283)
        - MonitoringClaimEvent(_submissionId,debridge.balance,IERC20Upgradeable(debridge.tokenAddress).totalSupply()) (contracts/transfers/DeBridgeGate.sol#961-965)
                - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
Reentrancy in DeBridgeGate.deployNewAsset(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)
        - deBridgeTokenAddress = IDeBridgeTokenDeployer(deBridgeTokenDeployer).deployAsset(debridgeId,_name,_symbol,_decimals) (contracts/transfers/DeBridgeGate.sol#334-335)
        Event emitted after the call(s):
        - PairAdded(_debridgeId,_tokenAddress,_nativeAddress,_nativeChainId,debridge.maxAmount,debridge.minReservesBps) (contracts/transfers/DeBridgeGate.sol#645-652)
                - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
Reentrancy in DeBridgeGate.flash(address,address,uint256,bytes) (contracts/transfers/DeBridgeGate.sol#287-308):
        External calls:
        - IERC20Upgradeable(_tokenAddress).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#299)
        - IFlashCallback(msg.sender).flashCallback(currentFlashFee,_data) (contracts/transfers/DeBridgeGate.sol#300)
        Event emitted after the call(s):
        - Flash(msg.sender,_tokenAddress,_receiver,_amount,paid) (contracts/transfers/DeBridgeGate.sol#307)
Reentrancy in DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes) (contracts/transfers/DeBridgeGate.sol#182-229):
        External calls:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - IERC20Permit(_tokenAddress).permit(msg.sender,address(this),_amount,deadline,v,r,s) (contracts/transfers/DeBridgeGate.sol#677-684)
                - (success,None) = _token.call(abi.encodeWithSignature(symbol())) (contracts/transfers/DeBridgeGate.sol#889)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
                - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/transfers/DeBridgeGate.sol#731)
                - IDeBridgeToken(debridge.tokenAddress).burn(amountAfterFee) (contracts/transfers/DeBridgeGate.sol#881)
        External calls sending eth:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
        Event emitted after the call(s):
        - MonitoringSendEvent(submissionId,nonce,getDebridge[_debridgeId].balance,IERC20Upgradeable(getDebridge[_debridgeId].tokenAddress).totalSupply()) (contracts/transfers/DeBridgeGate.sol#860-865)
                - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
        - Sent(submissionId,_debridgeId,_amount,_receiver,nonce,_chainIdTo,_referralCode,feeParams,abi.encode(autoParams),msg.sender) (contracts/transfers/DeBridgeGate.sol#846-857)
                - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
        - Sent(submissionId,_debridgeId,_amount,_receiver,nonce,_chainIdTo,_referralCode,feeParams,bytes(),msg.sender) (contracts/transfers/DeBridgeGate.sol#846-857)
                - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
Reentrancy in DeBridgeGate.withdrawFee(bytes32) (contracts/transfers/DeBridgeGate.sol#489-506):
        External calls:
        - _safeTransferETH(feeProxy,amount) (contracts/transfers/DeBridgeGate.sol#499)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        - IERC20Upgradeable(getDebridge[_debridgeId].tokenAddress).safeTransfer(feeProxy,amount) (contracts/transfers/DeBridgeGate.sol#503)
        External calls sending eth:
        - _safeTransferETH(feeProxy,amount) (contracts/transfers/DeBridgeGate.sol#499)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        Event emitted after the call(s):
        - WithdrawnFee(_debridgeId,amount) (contracts/transfers/DeBridgeGate.sol#505)
Reference: https://github.com/crytic/slither/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.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#181-184)
SignatureUtil.parseSignature(bytes,uint256) (contracts/libraries/SignatureUtil.sol#32-49) uses assembly
        - INLINE ASM (contracts/libraries/SignatureUtil.sol#41-45)
SignatureUtil.toUint256(bytes,uint256) (contracts/libraries/SignatureUtil.sol#51-61) uses assembly
        - INLINE ASM (contracts/libraries/SignatureUtil.sol#58-60)
DeBridgeGate.getChainId() (contracts/transfers/DeBridgeGate.sol#1134-1138) uses assembly
        - INLINE ASM (contracts/transfers/DeBridgeGate.sol#1135-1137)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

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/IAccessControlUpgradeable.sol#4)
        - ^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/security/ReentrancyGuardUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - 0.8.7 (contracts/interfaces/ICallProxy.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeGate.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeToken.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeTokenDeployer.sol#2)
        - 0.8.7 (contracts/interfaces/IERC20Permit.sol#3)
        - 0.8.7 (contracts/interfaces/IFlashCallback.sol#2)
        - 0.8.7 (contracts/interfaces/ISignatureVerifier.sol#2)
        - 0.8.7 (contracts/interfaces/IWETH.sol#2)
        - 0.8.7 (contracts/interfaces/IWethGate.sol#2)
        - 0.8.7 (contracts/libraries/Flags.sol#2)
        - 0.8.7 (contracts/libraries/SignatureUtil.sol#2)
        - 0.8.7 (contracts/transfers/DeBridgeGate.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/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._setupRole(bytes32,bytes32) (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#203-207) is never used and should be removed
AddressUpgradeable.functionCall(address,bytes) (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#109-115) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#142-144) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161) 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
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
ERC20._burn(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#275-290) is never used and should be removed
ERC20._mint(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#252-262) is never used and should be removed
Flags.setFlag(uint256,uint256,bool) (contracts/libraries/Flags.sol#34-45) is never used and should be removed
PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is never used and should be removed
PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is never used and should be removed
SafeERC20Upgradeable.safeApprove(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#45-58) is never used and should be removed
SafeERC20Upgradeable.safeDecreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#69-80) is never used and should be removed
SafeERC20Upgradeable.safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#60-67) is never used and should be removed
SignatureUtil.getUnsignedMsg(bytes32) (contracts/libraries/SignatureUtil.sol#13-15) is never used and should be removed
SignatureUtil.splitSignature(bytes) (contracts/libraries/SignatureUtil.sol#19-30) is never used and should be removed
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#48-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/security/PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/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
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-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#58)
Low level call in AddressUpgradeable.functionCallWithValue(address,bytes,uint256,string) (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#132)
Low level call in AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#159)
Low level call in DeBridgeGate._validateToken(address) (contracts/transfers/DeBridgeGate.sol#878-891):
        - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
        - (success,None) = _token.call(abi.encodeWithSignature(symbol())) (contracts/transfers/DeBridgeGate.sol#889)
Low level call in DeBridgeGate._safeTransferETH(address,uint256) (contracts/transfers/DeBridgeGate.sol#988-991):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
Reference: https://github.com/crytic/slither/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 AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is not in mixedCase
Function ReentrancyGuardUpgradeable.__ReentrancyGuard_init() (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#40-42) is not in mixedCase
Function ReentrancyGuardUpgradeable.__ReentrancyGuard_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#44-46) is not in mixedCase
Variable ReentrancyGuardUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#68) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 Flags.getFlag(uint256,uint256)._packedFlags (contracts/libraries/Flags.sol#26) is not in mixedCase
Parameter Flags.getFlag(uint256,uint256)._flag (contracts/libraries/Flags.sol#26) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._packedFlags (contracts/libraries/Flags.sol#37) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._flag (contracts/libraries/Flags.sol#38) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._value (contracts/libraries/Flags.sol#39) 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#19) is not in mixedCase
Parameter SignatureUtil.parseSignature(bytes,uint256)._signatures (contracts/libraries/SignatureUtil.sol#32) is not in mixedCase
Parameter SignatureUtil.toUint256(bytes,uint256)._bytes (contracts/libraries/SignatureUtil.sol#51) is not in mixedCase
Parameter SignatureUtil.toUint256(bytes,uint256)._offset (contracts/libraries/SignatureUtil.sol#51) is not in mixedCase
Parameter DeBridgeGate.initialize(uint8,IWETH)._excessConfirmations (contracts/transfers/DeBridgeGate.sol#169) is not in mixedCase
Parameter DeBridgeGate.initialize(uint8,IWETH)._weth (contracts/transfers/DeBridgeGate.sol#170) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._tokenAddress (contracts/transfers/DeBridgeGate.sol#183) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._amount (contracts/transfers/DeBridgeGate.sol#184) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._chainIdTo (contracts/transfers/DeBridgeGate.sol#185) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._receiver (contracts/transfers/DeBridgeGate.sol#186) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._permit (contracts/transfers/DeBridgeGate.sol#187) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._assetFeeId (contracts/transfers/DeBridgeGate.sol#188) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._referralCode (contracts/transfers/DeBridgeGate.sol#189) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._autoParams (contracts/transfers/DeBridgeGate.sol#190) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._debridgeId (contracts/transfers/DeBridgeGate.sol#233) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._amount (contracts/transfers/DeBridgeGate.sol#234) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._chainIdFrom (contracts/transfers/DeBridgeGate.sol#235) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._receiver (contracts/transfers/DeBridgeGate.sol#236) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._nonce (contracts/transfers/DeBridgeGate.sol#237) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._signatures (contracts/transfers/DeBridgeGate.sol#238) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._autoParams (contracts/transfers/DeBridgeGate.sol#239) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._tokenAddress (contracts/transfers/DeBridgeGate.sol#288) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._receiver (contracts/transfers/DeBridgeGate.sol#289) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._amount (contracts/transfers/DeBridgeGate.sol#290) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._data (contracts/transfers/DeBridgeGate.sol#291) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._nativeTokenAddress (contracts/transfers/DeBridgeGate.sol#318) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._nativeChainId (contracts/transfers/DeBridgeGate.sol#319) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._name (contracts/transfers/DeBridgeGate.sol#320) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._symbol (contracts/transfers/DeBridgeGate.sol#321) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._decimals (contracts/transfers/DeBridgeGate.sol#322) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._signatures (contracts/transfers/DeBridgeGate.sol#323) is not in mixedCase
Parameter DeBridgeGate.autoUpdateFixedNativeFee(uint256)._globalFixedNativeFee (contracts/transfers/DeBridgeGate.sol#343) is not in mixedCase
Parameter DeBridgeGate.updateChainSupport(uint256[],IDeBridgeGate.ChainSupportInfo[],bool)._chainIds (contracts/transfers/DeBridgeGate.sol#356) is not in mixedCase
Parameter DeBridgeGate.updateChainSupport(uint256[],IDeBridgeGate.ChainSupportInfo[],bool)._chainSupportInfo (contracts/transfers/DeBridgeGate.sol#357) is not in mixedCase
Parameter DeBridgeGate.updateChainSupport(uint256[],IDeBridgeGate.ChainSupportInfo[],bool)._isChainFrom (contracts/transfers/DeBridgeGate.sol#358) is not in mixedCase
Parameter DeBridgeGate.updateGlobalFee(uint256,uint16)._globalFixedNativeFee (contracts/transfers/DeBridgeGate.sol#376) is not in mixedCase
Parameter DeBridgeGate.updateGlobalFee(uint256,uint16)._globalTransferFeeBps (contracts/transfers/DeBridgeGate.sol#377) is not in mixedCase
Parameter DeBridgeGate.updateAssetFixedFees(bytes32,uint256[],uint256[])._debridgeId (contracts/transfers/DeBridgeGate.sol#389) is not in mixedCase
Parameter DeBridgeGate.updateAssetFixedFees(bytes32,uint256[],uint256[])._supportedChainIds (contracts/transfers/DeBridgeGate.sol#390) is not in mixedCase
Parameter DeBridgeGate.updateAssetFixedFees(bytes32,uint256[],uint256[])._assetFeesInfo (contracts/transfers/DeBridgeGate.sol#391) is not in mixedCase
Parameter DeBridgeGate.updateExcessConfirmations(uint8,_excessConfirmations) (contracts/transfers/DeBridgeGate.sol#402) is not in mixedCase
Parameter DeBridgeGate.setChainSupport(uint256,bool,bool)._chainId (contracts/transfers/DeBridgeGate.sol#411) is not in mixedCase
Parameter DeBridgeGate.setChainSupport(uint256,bool,bool)._isSupported (contracts/transfers/DeBridgeGate.sol#411) is not in mixedCase
Parameter DeBridgeGate.setChainSupport(uint256,bool,bool)._isChainFrom (contracts/transfers/DeBridgeGate.sol#411) is not in mixedCase
Parameter DeBridgeGate.setCallProxy(address)._callProxy (contracts/transfers/DeBridgeGate.sol#423) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._debridgeId (contracts/transfers/DeBridgeGate.sol#434) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._maxAmount (contracts/transfers/DeBridgeGate.sol#435) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._minReservesBps (contracts/transfers/DeBridgeGate.sol#436) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._amountThreshold (contracts/transfers/DeBridgeGate.sol#437) is not in mixedCase
Parameter DeBridgeGate.setSignatureVerifier(address)._verifier (contracts/transfers/DeBridgeGate.sol#458) is not in mixedCase
Parameter DeBridgeGate.setDeBridgeTokenDeployer(address)._deBridgeTokenDeployer (contracts/transfers/DeBridgeGate.sol#456) is not in mixedCase
Parameter DeBridgeGate.setDefiController(address)._defiController (contracts/transfers/DeBridgeGate.sol#462) is not in mixedCase
Parameter DeBridgeGate.setFeeContractUpdater(address)._value (contracts/transfers/DeBridgeGate.sol#468) is not in mixedCase
Parameter DeBridgeGate.setWethGate(IWethGate)._wethGate (contracts/transfers/DeBridgeGate.sol#474) is not in mixedCase
Parameter DeBridgeGate.withdrawFee(bytes32)._debridgeId (contracts/transfers/DeBridgeGate.sol#489) is not in mixedCase
Parameter DeBridgeGate.requestReserves(address,uint256)._tokenAddress (contracts/transfers/DeBridgeGate.sol#509) is not in mixedCase
Parameter DeBridgeGate.requestReserves(address,uint256)._amount (contracts/transfers/DeBridgeGate.sol#509) is not in mixedCase
Parameter DeBridgeGate.returnReserves(address,uint256)._tokenAddress (contracts/transfers/DeBridgeGate.sol#528) is not in mixedCase
Parameter DeBridgeGate.returnReserves(address,uint256)._amount (contracts/transfers/DeBridgeGate.sol#528) is not in mixedCase
Parameter DeBridgeGate.setFeeProxy(address)._feeProxy (contracts/transfers/DeBridgeGate.sol#547) is not in mixedCase
Parameter DeBridgeGate.blockSubmission(bytes32[],bool)._submissionIds (contracts/transfers/DeBridgeGate.sol#554) is not in mixedCase
Parameter DeBridgeGate.updateFlashFee(uint256)._flashFeeBps (contracts/transfers/DeBridgeGate.sol#567) is not in mixedCase
Parameter DeBridgeGate.updateFeeDiscount(address,uint16,uint16)._address (contracts/transfers/DeBridgeGate.sol#577) is not in mixedCase
Parameter DeBridgeGate.updateFeeDiscount(address,uint16,uint16)._discountFixBps (contracts/transfers/DeBridgeGate.sol#578) is not in mixedCase
Parameter DeBridgeGate.updateFeeDiscount(address,uint16,uint16)._discountTransferBps (contracts/transfers/DeBridgeGate.sol#579) is not in mixedCase
Parameter DeBridgeGate.getDefiAvailableReserves(address)._tokenAddress (contracts/transfers/DeBridgeGate.sol#1029) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,address)._chainId (contracts/transfers/DeBridgeGate.sol#1042) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,address)._tokenAddress (contracts/transfers/DeBridgeGate.sol#1042) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,bytes)._chainId (contracts/transfers/DeBridgeGate.sol#1049) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,bytes)._tokenAddress (contracts/transfers/DeBridgeGate.sol#1049) is not in mixedCase
Parameter DeBridgeGate.getDebridgeChainAssetFixedFee(bytes32,uint256)._debridgeId (contracts/transfers/DeBridgeGate.sol#1055) is not in mixedCase
Parameter DeBridgeGate.getDebridgeChainAssetFixedFee(bytes32,uint256)._chainId (contracts/transfers/DeBridgeGate.sol#1056) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._debridgeId (contracts/transfers/DeBridgeGate.sol#1072) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._chainIdFrom (contracts/transfers/DeBridgeGate.sol#1073) is not in mixedCase
```

```
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)._autoParams (contracts/transfers/DeBridgeGate.sol#1077) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._hasAutoParams (contracts/transfers/DeBridgeGate.sol#1078) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._sender (contracts/transfers/DeBridgeGate.sol#1079) 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)._symbol (contracts/transfers/DeBridgeGate.sol#1122) is not in mixedCase
Parameter DeBridgeGate.getDeployId(bytes32,string,string,uint8)._decimals (contracts/transfers/DeBridgeGate.sol#1123) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

ReentrancyGuardUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#68) is never used in DeBridgeGate (contracts/transfers/DeBridgeGate.sol#26-1145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DeBridgeGate.lockedClaim (contracts/transfers/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/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#170-174)
name() should be declared external:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#62-64)
symbol() should be declared external:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#70-72)
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,IWETH) should be declared external:
        - DeBridgeGate.initialize(uint8,IWETH) (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/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#31)
PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/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/utils/ContextUpgradeable.sol#31)
ERC20PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) shadows:
        - PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97)
        - ERC20Upgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#362)
        - 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_
rade.sol#71)
ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#80-108) ignores return value by Address.functionDelegateCall(newImplementation,data)
1967Upgrade.sol#90)
ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#80-108) ignores return value by Address.functionDelegateCall(newImplementation,abi.e
) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#98-101)
ERC1967Upgrade._upgradeBeaconToAndCall(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:
        - debridgeAddress = _debridgeAddress (contracts/transfers/DeBridgeTokenDeployer.sol#75)
DeBridgeTokenDeployer.setDebridgeAddress(address) (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,address)._deBridgeTokenAdmin (contracts/transfers/DeBridgeTokenDeployer.sol#70) lacks a zero-check on :
        - deBridgeTokenAdmin = _deBridgeTokenAdmin (contracts/transfers/DeBridgeTokenDeployer.sol#74)
DeBridgeTokenDeployer.initialize(address,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 calls:
        - Address.functionDelegateCall(newImplementation,data) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#90)
        - Address.functionDelegateCall(newImplementation,abi.encodeWithSignature(upgradeTo(address),oldImplementation)) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#98-101)
        Event emitted after the call(s):
        - Upgraded(newImplementation) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#56)
                - _upgradeTo(newImplementation) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#106)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,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.sol#22-45) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/proxy/Proxy.sol#23-44)
Address.isContract(address) (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.getAddressSlot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#52-56) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#53-55)
StorageSlot.getBooleanSlot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#61-65) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#62-64)
StorageSlot.getBytes32Slot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#70-74) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#71-73)
StorageSlot.getUint256Slot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#79-83) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#80-82)
DeBridgeToken.initialize(string,string,uint8,address,address[]) (contracts/periphery/DeBridgeToken.sol#54-90) uses assembly
        - INLINE ASM (contracts/periphery/DeBridgeToken.sol#76-78)
DeBridgeTokenDeployer.deployAsset(bytes32,string,string,uint8) (contracts/transfers/DeBridgeTokenDeployer.sol#85-138) uses assembly
        - INLINE ASM (contracts/transfers/DeBridgeTokenDeployer.sol#122-129)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
```

```
Different versions of Solidity is used:
        - Version used: ['0.8.7', '^0.8.0', '^0.8.2']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/access/IAccessControlUpgradeable.sol#4)
        - ^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/token/ERC20/ERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - ^0.8.2 (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/proxy/Proxy.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/proxy/beacon/BeaconProxy.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/proxy/beacon/IBeacon.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#4)
        - 0.8.7 (contracts/interfaces/IDeBridgeToken.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeTokenDeployer.sol#2)
        - 0.8.7 (contracts/interfaces/IERC20Permit.sol#3)
        - 0.8.7 (contracts/periphery/DeBridgeToken.sol#2)
        - 0.8.7 (contracts/periphery/DeBridgeTokenProxy.sol#2)
        - 0.8.7 (contracts/transfers/DeBridgeTokenDeployer.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/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
Address.functionCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#80-82) is never used and should be removed
Address.functionCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#90-96) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#109-115) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#142-144) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#55-60) is never used and should be removed
BeaconProxy._beacon() (node_modules/@openzeppelin/contracts/proxy/beacon/BeaconProxy.sol#38-40) is never used and should be removed
BeaconProxy._setBeacon(address,bytes) (node_modules/@openzeppelin/contracts/proxy/beacon/BeaconProxy.sol#59-61) is never used and should be removed
ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
ERC1967Upgrade._changeAdmin(address) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#142-145) is never used and should be removed
ERC1967Upgrade._getAdmin() (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#125-127) is never used and should be removed
ERC1967Upgrade._getImplementation() (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#37-39) is never used and should be removed
ERC1967Upgrade._setAdmin(address) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#132-135) is never used and should be removed
ERC1967Upgrade._setImplementation(address) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#44-47) is never used and should be removed
ERC1967Upgrade._upgradeTo(address) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#54-57) is never used and should be removed
ERC1967Upgrade._upgradeToAndCall(address,bytes,bool) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#64-73) is never used and should be removed
ERC1967Upgrade._upgradeToAndCallSecure(address,bytes,bool) (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#88-108) is never used and should be removed
ERC20PausableUpgradeable.__ERC20Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#18-22) is never used and should be removed
ERC20PausableUpgradeable.__ERC20Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#24-25) is never used and should be removed
ERC20Upgradeable.__ERC20_init(string,string) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#55-58) is never used and should be removed
PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is never used and should be removed
PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is never used and should be removed
Proxy._implementation() (node_modules/@openzeppelin/contracts/proxy/Proxy.sol#51) is never used and should be removed
StorageSlot.getBooleanSlot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#61-65) is never used and should be removed
StorageSlot.getBytes32Slot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#78-74) is never used and should be removed
StorageSlot.getUint256Slot(bytes32) (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#79-83) is never used and should be removed
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/security/PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.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/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
Pragma version^0.8.2 (node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/proxy/Proxy.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/proxy/beacon/BeaconProxy.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/proxy/beacon/IBeacon.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/StorageSlot.sol#4) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#55-60):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#58)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#123-134):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#132)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#159)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#179-188):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#186)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

DeBridgeTokenDeployer (contracts/transfers/DeBridgeTokenDeployer.sol#12-187) should inherit from IBeacon (node_modules/@openzeppelin/contracts/proxy/beacon/IBeacon.sol#9-16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-inheritance

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 AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is not in mixedCase
Function ERC20Upgradeable.__ERC20_init(string,string) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#55-58) is not in mixedCase
Function ERC20Upgradeable.__ERC20_init_unchained(string,string) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#60-63) is not in mixedCase
Variable ERC20Upgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#362) is not in mixedCase
Function ERC20PausableUpgradeable.__ERC20Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#18-22) is not in mixedCase
Function ERC20PausableUpgradeable.__ERC20Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#24-25) is not in mixedCase
Variable ERC20PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 DeBridgeToken.mint(address,uint256)._receiver (contracts/periphery/DeBridgeToken.sol#93) is not in mixedCase
Parameter DeBridgeToken.mint(address,uint256)._amount (contracts/periphery/DeBridgeToken.sol#93) is not in mixedCase
Parameter DeBridgeToken.burn(uint256)._amount (contracts/periphery/DeBridgeToken.sol#98) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._owner (contracts/periphery/DeBridgeToken.sol#111) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._spender (contracts/periphery/DeBridgeToken.sol#112) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._value (contracts/periphery/DeBridgeToken.sol#113) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._deadline (contracts/periphery/DeBridgeToken.sol#114) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._v (contracts/periphery/DeBridgeToken.sol#115) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._r (contracts/periphery/DeBridgeToken.sol#116) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._s (contracts/periphery/DeBridgeToken.sol#117) is not in mixedCase
Variable DeBridgeToken.DOMAIN_SEPARATOR (contracts/periphery/DeBridgeToken.sol#21) is not in mixedCase
Variable DeBridgeToken._decimals (contracts/periphery/DeBridgeToken.sol#29) is not in mixedCase
Parameter DeBridgeTokenDeployer.initialize(address,address,address)._tokenImplementation (contracts/transfers/DeBridgeTokenDeployer.sol#69) is not in mixedCase
Parameter DeBridgeTokenDeployer.initialize(address,address,address)._deBridgeTokenAdmin (contracts/transfers/DeBridgeTokenDeployer.sol#70) is not in mixedCase
Parameter DeBridgeTokenDeployer.initialize(address,address,address)._debridgeAddress (contracts/transfers/DeBridgeTokenDeployer.sol#71) is not in mixedCase
Parameter DeBridgeTokenDeployer.deployAsset(bytes32,string,string,uint8)._debridgeId (contracts/transfers/DeBridgeTokenDeployer.sol#86) is not in mixedCase
Parameter DeBridgeTokenDeployer.deployAsset(bytes32,string,string,uint8)._name (contracts/transfers/DeBridgeTokenDeployer.sol#87) is not in mixedCase
Parameter DeBridgeTokenDeployer.deployAsset(bytes32,string,string,uint8)._symbol (contracts/transfers/DeBridgeTokenDeployer.sol#88) is not in mixedCase
Parameter DeBridgeTokenDeployer.deployAsset(bytes32,string,string,uint8)._decimals (contracts/transfers/DeBridgeTokenDeployer.sol#89) is not in mixedCase
Parameter DeBridgeTokenDeployer.setTokenImplementation(address)._impl (contracts/transfers/DeBridgeTokenDeployer.sol#150) is not in mixedCase
Parameter DeBridgeTokenDeployer.setDeBridgeTokenAdmin(address)._deBridgeTokenAdmin (contracts/transfers/DeBridgeTokenDeployer.sol#157) is not in mixedCase
Parameter DeBridgeTokenDeployer.setDebridgeAddress(address)._debridgeAddress (contracts/transfers/DeBridgeTokenDeployer.sol#164) is not in mixedCase
Parameter DeBridgeTokenDeployer.setOverridedTokenInfo(bytes32[],DeBridgeTokenDeployer.OverridedTokenInfo[])._debridgeIds (contracts/transfers/DeBridgeTokenDeployer.sol#173) is not in mixedCase
Parameter DeBridgeTokenDeployer.setOverridedTokenInfo(bytes32[],DeBridgeTokenDeployer.OverridedTokenInfo[])._tokens (contracts/transfers/DeBridgeTokenDeployer.sol#174) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
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#10-166)
AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is never used in DeBridgeTokenDeployer (contracts/transfers/DeBridgeTokenDeployer.sol#12-187)
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#170-174)
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)
approve(address,uint256) should be declared external:
        - ERC20Upgradeable.approve(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)
implementation() should be declared external:
        - DeBridgeTokenDeployer.implementation() (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#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)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

Different versions of Solidity is used:
        - Version used: ['0.8.7', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/access/IAccessControlUpgradeable.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#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/IERC165Upgradeable.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-pragma-directives-are-used

OraclesManager.addOracles(address[],bool[]) (contracts/transfers/OraclesManager.sol#78-101) has costly operations inside a loop:
        - requiredOraclesCount += 1 (contracts/transfers/OraclesManager.sol#92)
OraclesManager.updateOracle(address,bool,bool) (contracts/transfers/OraclesManager.sol#107-139) has costly operations inside a loop:
        - oracleAddresses.pop() (contracts/transfers/OraclesManager.sol#128)
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#51-55) 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
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/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
OraclesManager.initialize(uint8,uint8) (contracts/transfers/OraclesManager.sol#52-57) is never used and should be removed
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/utils/ContextUpgradeable.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#51-55) is not in mixedCase
Function AccessControlUpgradeable.__AccessControl_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-58) is not in mixedCase
Variable AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 OraclesManager.initialize(uint8,uint8)._minConfirmations (contracts/transfers/OraclesManager.sol#52) is not in mixedCase
Parameter OraclesManager.initialize(uint8,uint8)._excessConfirmations (contracts/transfers/OraclesManager.sol#52) is not in mixedCase
Parameter OraclesManager.setMinConfirmations(uint8)._minConfirmations (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[],bool[])._oracles (contracts/transfers/OraclesManager.sol#79) is not in mixedCase
Parameter OraclesManager.addOracles(address[],bool[])._required (contracts/transfers/OraclesManager.sol#80) is not in mixedCase
Parameter OraclesManager.updateOracle(address,bool,bool)._oracle (contracts/transfers/OraclesManager.sol#108) is not in mixedCase
Parameter OraclesManager.updateOracle(address,bool,bool)._isValid (contracts/transfers/OraclesManager.sol#109) is not in mixedCase
Parameter OraclesManager.updateOracle(address,bool,bool)._required (contracts/transfers/OraclesManager.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#232) is never used in OraclesManager (contracts/transfers/OraclesManager.sol#10-140)
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) shadows:
        - ERC165Upgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#36)
        - ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

SignatureVerifier.submit(bytes32,bytes,uint8) (contracts/transfers/SignatureVerifier.sol#58-114) 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).currentRequiredOraclesCount (contracts/transfers/SignatureVerifier.sol#68) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-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)._debridgeAddress (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) (contracts/libraries/SignatureUtil.sol#32-49) uses assembly
        - INLINE ASM (contracts/libraries/SignatureUtil.sol#41-45)
SignatureUtil.toUint256(bytes,uint256) (contracts/libraries/SignatureUtil.sol#51-61) uses assembly
        - INLINE ASM (contracts/libraries/SignatureUtil.sol#58-60)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

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/IAccessControlUpgradeable.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#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - 0.8.7 (contracts/interfaces/IOraclesManager.sol#2)
        - 0.8.7 (contracts/interfaces/ISignatureVerifier.sol#2)
        - 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.addOracles(address[],bool[]) (contracts/transfers/OraclesManager.sol#78-101) has costly operations inside a loop:
        - requiredOraclesCount += 1 (contracts/transfers/OraclesManager.sol#92)
OraclesManager.updateOracle(address,bool,bool) (contracts/transfers/OraclesManager.sol#107-139) has costly operations inside a loop:
        - oracleAddresses.pop() (contracts/transfers/OraclesManager.sol#128)
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#51-55) 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
ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
SignatureUtil.toUint256(bytes,uint256) (contracts/libraries/SignatureUtil.sol#51-61) is never used and should be removed
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/utils/ContextUpgradeable.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#51-55) is not in mixedCase
Function AccessControlUpgradeable.__AccessControl_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-58) is not in mixedCase
Variable AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 SignatureUtil.getUnsignedMsg(bytes32)._submissionId (contracts/libraries/SignatureUtil.sol#13) is not in mixedCase
Parameter SignatureUtil.splitSignature(bytes)._signature (contracts/libraries/SignatureUtil.sol#19) is not in mixedCase
Parameter SignatureUtil.parseSignature(bytes,uint256)._signatures (contracts/libraries/SignatureUtil.sol#32) is not in mixedCase
Parameter SignatureUtil.toUint256(bytes,uint256)._bytes (contracts/libraries/SignatureUtil.sol#51) is not in mixedCase
Parameter SignatureUtil.toUint256(bytes,uint256)._offset (contracts/libraries/SignatureUtil.sol#51) is not in mixedCase
Parameter OraclesManager.initialize(uint8,uint8)._minConfirmations (contracts/transfers/OraclesManager.sol#52) is not in mixedCase
Parameter OraclesManager.initialize(uint8,uint8)._excessConfirmations (contracts/transfers/OraclesManager.sol#52) is not in mixedCase
Parameter OraclesManager.setMinConfirmations(uint8)._minConfirmations (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[],bool[])._oracles (contracts/transfers/OraclesManager.sol#79) is not in mixedCase
Parameter OraclesManager.addOracles(address[],bool[])._required (contracts/transfers/OraclesManager.sol#80) is not in mixedCase
Parameter OraclesManager.updateOracle(address,bool,bool)._oracle (contracts/transfers/OraclesManager.sol#108) is not in mixedCase
Parameter OraclesManager.updateOracle(address,bool,bool)._isValid (contracts/transfers/OraclesManager.sol#109) is not in mixedCase
Parameter OraclesManager.updateOracle(address,bool,bool)._required (contracts/transfers/OraclesManager.sol#110) is not in mixedCase
Parameter SignatureVerifier.initialize(uint8,uint8,uint8,address)._minConfirmations (contracts/transfers/SignatureVerifier.sol#46) is not in mixedCase
Parameter SignatureVerifier.initialize(uint8,uint8,uint8,address)._confirmationThreshold (contracts/transfers/SignatureVerifier.sol#47) is not in mixedCase
Parameter SignatureVerifier.initialize(uint8,uint8,uint8,address)._excessConfirmations (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.submit(bytes32,bytes,uint8)._submissionId (contracts/transfers/SignatureVerifier.sol#59) is not in mixedCase
Parameter SignatureVerifier.submit(bytes32,bytes,uint8)._signatures (contracts/transfers/SignatureVerifier.sol#60) is not in mixedCase
Parameter SignatureVerifier.submit(bytes32,bytes,uint8)._excessConfirmations (contracts/transfers/SignatureVerifier.sol#61) is not in mixedCase
Parameter SignatureVerifier.setThreshold(uint8)._confirmationThreshold (contracts/transfers/SignatureVerifier.sol#120) is not in mixedCase
Parameter SignatureVerifier.setDebridgeAddress(address)._debridgeAddress (contracts/transfers/SignatureVerifier.sol#127) is not in mixedCase
Parameter SignatureVerifier.isValidSignature(bytes32,bytes)._submissionId (contracts/transfers/SignatureVerifier.sol#136) 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/AccessControlUpgradeable.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)
renounceRole(bytes32,address) should be declared external:
        - AccessControlUpgradeable.renounceRole(bytes32,address) (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.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,uint256) (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 calls:
        - weth.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 calls sending eth:
        - _safeTransferETH(_receiver,_wad) (contracts/transfers/WethGate.sol#33)
                - (success) = _to.call{value: _value}(new bytes(0)) (contracts/transfers/WethGate.sol#38)
        Event emitted after the call(s):
        - Withdrawal(_receiver,_wad) (contracts/transfers/WethGate.sol#34)
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
```

AUTOMATED TESTING

SimpleFeeProxy.sol

```
SimpleFeeProxy._safeTransferETH(address,uint256) (contracts/periphery/SimpleFeeProxy.sol#121-124) sends eth to arbitrary user
        Dangerous calls:
        - (success) = to.call{value: value}(new bytes(0)) (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/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#31)
PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
        - 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(IDeBridgeGate,address)._treasury (contracts/periphery/SimpleFeeProxy.sol#36) lacks a zero-check on :
                - treasury = _treasury (contracts/periphery/SimpleFeeProxy.sol#38)
SimpleFeeProxy.setTreasury(address)._treasury (contracts/periphery/SimpleFeeProxy.sol#56) lacks a zero-check on :
                - treasury = _treasury (contracts/periphery/SimpleFeeProxy.sol#57)
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#33-35)
AddressUpgradeable.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#181-184)
SimpleFeeProxy.getChainId() (contracts/periphery/SimpleFeeProxy.sol#110-114) uses assembly
        - INLINE ASM (contracts/periphery/SimpleFeeProxy.sol#111-113)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

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/IAccessControlUpgradeable.sol#4)
        - ^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/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - 0.8.7 (contracts/interfaces/IDeBridgeGate.sol#2)
        - 0.8.7 (contracts/periphery/SimpleFeeProxy.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/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.functionCall(address,bytes) (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#109-115) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#142-144) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161) 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/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is never used and should be removed
PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is never used and should be removed
SafeERC20Upgradeable.safeApprove(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#45-58) is never used and should be removed
SafeERC20Upgradeable.safeDecreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#69-80) is never used and should be removed
SafeERC20Upgradeable.safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#60-67) is never used and should be removed
SafeERC20Upgradeable.safeTransferFrom(IERC20Upgradeable,address,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#29-36) is never used and should be remov
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/security/PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/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

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#58)
Low level call in AddressUpgradeable.functionCallWithValue(address,bytes,uint256,string) (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#132)
Low level call in AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#159)
Low level call in SimpleFeeProxy._safeTransferETH(address,uint256) (contracts/periphery/SimpleFeeProxy.sol#121-124):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/periphery/SimpleFeeProxy.sol#122)
Reference: https://github.com/crytic/slither/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 AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 SimpleFeeProxy.initialize(IDeBridgeGate,address)._debridgeGate (contracts/periphery/SimpleFeeProxy.sol#36) is not in mixedCase
Parameter SimpleFeeProxy.initialize(IDeBridgeGate,address)._treasury (contracts/periphery/SimpleFeeProxy.sol#36) is not in mixedCase
Parameter SimpleFeeProxy.setDebridgeGate(IDeBridgeGate)._debridgeGate (contracts/periphery/SimpleFeeProxy.sol#52) is not in mixedCase
Parameter SimpleFeeProxy.setTreasury(address)._treasury (contracts/periphery/SimpleFeeProxy.sol#56) is not in mixedCase
Parameter SimpleFeeProxy.withdrawFee(address)._tokenAddress (contracts/periphery/SimpleFeeProxy.sol#62) is not in mixedCase
Parameter SimpleFeeProxy.getDebridgeId(uint256,bytes)._chainId (contracts/periphery/SimpleFeeProxy.sol#94) is not in mixedCase
Parameter SimpleFeeProxy.getDebridgeId(uint256,bytes)._tokenAddress (contracts/periphery/SimpleFeeProxy.sol#94) is not in mixedCase
Parameter SimpleFeeProxy.getDebridgeId(uint256,address)._chainId (contracts/periphery/SimpleFeeProxy.sol#105) is not in mixedCase
Parameter SimpleFeeProxy.getDebridgeId(uint256,address)._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(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#170-174)
initialize(IDeBridgeGate,address) should be declared external:
        - SimpleFeeProxy.initialize(IDeBridgeGate,address) (contracts/periphery/SimpleFeeProxy.sol#36-40)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

DefiController.sol

```
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#31)
PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
        - ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

Reentrancy in DefiController.depositToStrategy(uint256,address) (contracts/periphery/DefiController.sol#91-117):
        External calls:
        - deBridgeGate.requestReserves(strategy.stakeToken,_amount) (contracts/periphery/DefiController.sol#109)
        - IERC20Upgradeable(strategy.stakeToken).safeApprove(address(strategyController),0) (contracts/periphery/DefiController.sol#112)
        - IERC20Upgradeable(strategy.stakeToken).safeApprove(address(strategyController),_amount) (contracts/periphery/DefiController.sol#113)
        - strategyController.deposit(strategy.stakeToken,_amount) (contracts/periphery/DefiController.sol#114)
        Event emitted after the call(s):
        - DepositToStrategy(_strategy,_amount) (contracts/periphery/DefiController.sol#116)
Reentrancy in DefiController.withdrawFromStrategy(uint256,address) (contracts/periphery/DefiController.sol#119-136):
        External calls:
        - strategyController.withdraw(strategy.strategyToken,_amount) (contracts/periphery/DefiController.sol#126)
        - IERC20Upgradeable(strategy.stakeToken).safeApprove(address(deBridgeGate),0) (contracts/periphery/DefiController.sol#127)
        - IERC20Upgradeable(strategy.stakeToken).safeApprove(address(deBridgeGate),_amount) (contracts/periphery/DefiController.sol#128)
        - deBridgeGate.returnReserves(strategy.stakeToken,_amount) (contracts/periphery/DefiController.sol#133)
        Event emitted after the call(s):
        - WithdrawFromStrategy(_strategy,_amount) (contracts/periphery/DefiController.sol#135)
Reference: https://github.com/crytic/slither/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.verifyCallResult(bool,bytes,string) (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/slither/wiki/Detector-Documentation#assembly-usage

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/IAccessControlUpgradeable.sol#4)
        - ^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/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - 0.8.7 (contracts/interfaces/IDeBridgeGate.sol#2)
        - 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/periphery/DefiController.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/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.functionCall(address,bytes) (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#109-115) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#142-144) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161) 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/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is never used and should be removed
PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is never used and should be removed
SafeERC20Upgradeable.safeDecreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#69-80) is never used and should be removed
SafeERC20Upgradeable.safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#60-67) is never used and should be removed
SafeERC20Upgradeable.safeTransfer(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#21-27) is never used and should be removed
SafeERC20Upgradeable.safeTransferFrom(IERC20Upgradeable,address,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#29-36) is never used and should be remov
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/security/PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/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

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#58)
Low level call in AddressUpgradeable.functionCallWithValue(address,bytes,uint256,string) (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#132)
Low level call in AddressUpgradeable.functionStaticCall(address,bytes,string) (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/slither/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 AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/interfaces/IUniswapV2Pair.sol#30) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/interfaces/IUniswapV2Pair.sol#32) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/interfaces/IUniswapV2Pair.sol#58) is not in mixedCase
Parameter DefiController.depositToStrategy(uint256,address)._amount (contracts/periphery/DefiController.sol#91) 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)._strategy (contracts/periphery/DefiController.sol#119) is not in mixedCase
Parameter DefiController.rebalanceStrategy(address)._strategy (contracts/periphery/DefiController.sol#138) is not in mixedCase
Parameter DefiController.isStrategyUnbalanced(address)._strategy (contracts/periphery/DefiController.sol#156) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address,address)._strategy (contracts/periphery/DefiController.sol#205) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address,address)._isEnabled (contracts/periphery/DefiController.sol#206) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address,address)._maxReservesBps (contracts/periphery/DefiController.sol#207) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address,address)._stakeToken (contracts/periphery/DefiController.sol#208) is not in mixedCase
Parameter DefiController.addStrategy(address,bool,uint16,address,address)._strategyToken (contracts/periphery/DefiController.sol#209) is not in mixedCase
Parameter DefiController.updateStrategy(address,bool,uint16)._strategy (contracts/periphery/DefiController.sol#240) is not in mixedCase
Parameter DefiController.updateStrategy(address,bool,uint16)._isEnabled (contracts/periphery/DefiController.sol#241) is not in mixedCase
Parameter DefiController.updateStrategy(address,bool,uint16)._maxReservesBps (contracts/periphery/DefiController.sol#242) is not in mixedCase
Parameter DefiController.setDeBridgeGate(IDeBridgeGate)._deBridgeGate (contracts/periphery/DefiController.sol#265) is not in mixedCase
Parameter DefiController.addWorker(address)._worker (contracts/periphery/DefiController.sol#269) is not in mixedCase
Parameter DefiController.removeWorker(address)._worker (contracts/periphery/DefiController.sol#273) 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 DefiController (contracts/periphery/DefiController.sol#14-291)
Reference: https://github.com/crytic/slither/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#170-174)
initialize() should be declared external:
        - DefiController.initialize() (contracts/periphery/DefiController.sol#82-89)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

CallProxy.sol

```
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#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) performs a multiplication on the result of a division:
        -sstore(uint256,uint256)(_preBytes,fslot_concatStorage_asm_0 + mload(uint256)(_postBytes + 0x20) / 0x100 ** 32 - mlength_concatStorage_asm_0 * 0x100 ** 32 - newlength_concatStorage_asm_0 + mlength_concatSto
0)
BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) performs a multiplication on the result of a division:
        -sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#223)
BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) performs a multiplication on the result of a division:
        -sstore(uint256,uint256)(sc_concatStorage_asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#189)
BytesLib.equalStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#439-589) performs a multiplication on the result of a division:
        -fslot_equalStorage_asm_0 = fslot_equalStorage_asm_0 / 0x100 * 0x100 (contracts/libraries/BytesLib.sol#466)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

CallProxy.call(address,address,bytes,uint256,bytes,uint256)._reserveAddress (contracts/periphery/CallProxy.sol#71) lacks a zero-check on :
        - (success) = _reserveAddress.call{value: amount}(new bytes(0)) (contracts/periphery/CallProxy.sol#95)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
uint256)(0x0,0x20) + slength_concatStorage_asm_0 / 32 (contracts/libraries/BytesLib.sol#195)
Variable 'BytesLib.concatStorage(bytes,bytes).submod_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#161)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used bef
thmod_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#204)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
d_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#205)
Variable 'BytesLib.concatStorage(bytes,bytes).submod_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#161)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used bef
ubmod_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#205)
Variable 'BytesLib.concatStorage(bytes,bytes).end_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#163)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
ngth_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#206)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used befor
d_concatStorage_asm_0 - 1 (contracts/libraries/BytesLib.sol#207)
Variable 'BytesLib.concatStorage(bytes,bytes).submod_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#161)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used bef
mod_concatStorage_asm_0 - 1 (contracts/libraries/BytesLib.sol#207)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
m_0,sload(uint256)(sc_concatStorage_asm_0) + mload(uint256)(mc_concatStorage_asm_0) & mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#209)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
m_0,sload(uint256)(sc_concatStorage_asm_0) + mload(uint256)(mc_concatStorage_asm_0) & mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#209)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used befor
asm_0,sload(uint256)(sc_concatStorage_asm_0) + mload(uint256)(mc_concatStorage_asm_0) & mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#209)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
sm_0 + 1 (contracts/libraries/BytesLib.sol#212)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
sm_0 + 0x20 (contracts/libraries/BytesLib.sol#213)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
asm_0 (contracts/libraries/BytesLib.sol#214)
Variable 'BytesLib.concatStorage(bytes,bytes).end_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#163)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
_asm_0 (contracts/libraries/BytesLib.sol#214)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used befor
ncatStorage_asm_0 - end_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#221)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
atStorage_asm_0 - end_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#221)
Variable 'BytesLib.concatStorage(bytes,bytes).end_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#163)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
catStorage_asm_0 - end_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#221)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
m_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#223)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
m_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#223)
Variable 'BytesLib.concatStorage(bytes,bytes).mask_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#164)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used befor
asm_0,mload(uint256)(mc_concatStorage_asm_0) / mask_concatStorage_asm_0 * mask_concatStorage_asm_0) (contracts/libraries/BytesLib.sol#223)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
m_0,mload(uint256)(mc_concatStorage_asm_0)) (contracts/libraries/BytesLib.sol#218)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
m_0,mload(uint256)(mc_concatStorage_asm_0)) (contracts/libraries/BytesLib.sol#218)
Variable 'BytesLib.concatStorage(bytes,bytes).sc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#147)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
sm_0 + 1 (contracts/libraries/BytesLib.sol#215)
Variable 'BytesLib.concatStorage(bytes,bytes).mc_concatStorage_asm_0 (contracts/libraries/BytesLib.sol#162)' in BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) potentially used before
sm_0 + 0x20 (contracts/libraries/BytesLib.sol#216)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

Reentrancy in CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256) (contracts/periphery/CallProxy.sol#101-134):
        External calls:
        - _customApprove(IERC20Upgradeable(_token),_receiver,amount) (contracts/periphery/CallProxy.sol#112)
                - returndata = address(token).functionCall(abi.encodeWithSelector(token.approve.selector,spender,value),ERC20 approve failed) (contracts/periphery/CallProxy.sol#210-213)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        External calls sending eth:
        - _customApprove(IERC20Upgradeable(_token),_receiver,amount) (contracts/periphery/CallProxy.sol#112)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        State variables written after the call(s):
        - _result = _externalCall(_receiver,0,_data,_nativeSender,_chainIdFrom,_flags) (contracts/periphery/CallProxy.sol#114-121)
                - submissionChainIdFrom = _chainIdFrom (contracts/periphery/CallProxy.sol#173)
                - submissionChainIdFrom = 0 (contracts/periphery/CallProxy.sol#204)
        - _result = _externalCall(_receiver,0,_data,_nativeSender,_chainIdFrom,_flags) (contracts/periphery/CallProxy.sol#114-121)
                - submissionNativeSender = _nativeSender (contracts/periphery/CallProxy.sol#174)
                - submissionNativeSender =  (contracts/periphery/CallProxy.sol#205)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

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.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#181-184)
BytesLib.concat(bytes,bytes) (contracts/libraries/BytesLib.sol#13-89) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#23-86)
BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#92-225)
BytesLib.slice(bytes,uint256,uint256) (contracts/libraries/BytesLib.sol#228-295) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#242-292)
BytesLib.toAddress(bytes,uint256) (contracts/libraries/BytesLib.sol#297-306) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#301-303)
BytesLib.toUint8(bytes,uint256) (contracts/libraries/BytesLib.sol#308-317) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#312-314)
BytesLib.toUint16(bytes,uint256) (contracts/libraries/BytesLib.sol#319-328) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#323-325)
BytesLib.toUint32(bytes,uint256) (contracts/libraries/BytesLib.sol#330-339) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#334-336)
BytesLib.toUint64(bytes,uint256) (contracts/libraries/BytesLib.sol#341-350) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#345-347)
BytesLib.toUint96(bytes,uint256) (contracts/libraries/BytesLib.sol#352-361) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#356-358)
BytesLib.toUint128(bytes,uint256) (contracts/libraries/BytesLib.sol#363-372) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#367-369)
BytesLib.toUint256(bytes,uint256) (contracts/libraries/BytesLib.sol#374-383) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#378-380)
BytesLib.toBytes32(bytes,uint256) (contracts/libraries/BytesLib.sol#385-394) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#389-391)
BytesLib.equal(bytes,bytes) (contracts/libraries/BytesLib.sol#396-437) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#399-434)
BytesLib.equalStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#439-589) uses assembly
        - INLINE ASM (contracts/libraries/BytesLib.sol#449-586)
MultiSendCallOnly._multiSend(bytes) (contracts/libraries/MultiSendCallOnly.sol#21-60) uses assembly
        - INLINE ASM (contracts/libraries/MultiSendCallOnly.sol#23-59)
CallProxy._externalCall(address,uint256,bytes,bytes,uint256,uint256) (contracts/periphery/CallProxy.sol#159-207) uses assembly
        - INLINE ASM (contracts/periphery/CallProxy.sol#193-196)
        - INLINE ASM (contracts/periphery/CallProxy.sol#198-200)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.8.7', '>=0.7.0<0.9.0', '>=0.8.0<0.9.0', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/access/IAccessControlUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - 0.8.7 (contracts/interfaces/ICallProxy.sol#2)
        - >=0.8.0<0.9.0 (contracts/libraries/BytesLib.sol#9)
        - 0.8.7 (contracts/libraries/Flags.sol#2)
        - >=0.7.0<0.9.0 (contracts/libraries/MultiSendCallOnly.sol#2)
        - 0.8.7 (contracts/periphery/CallProxy.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/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.functionCall(address,bytes) (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#109-115) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#142-144) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161) 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
BytesLib.concat(bytes,bytes) (contracts/libraries/BytesLib.sol#13-89) is never used and should be removed
BytesLib.concatStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#91-226) is never used and should be removed
BytesLib.equal(bytes,bytes) (contracts/libraries/BytesLib.sol#396-437) is never used and should be removed
BytesLib.equalStorage(bytes,bytes) (contracts/libraries/BytesLib.sol#439-589) is never used and should be removed
BytesLib.toAddress(bytes,uint256) (contracts/libraries/BytesLib.sol#297-306) is never used and should be removed
BytesLib.toBytes32(bytes,uint256) (contracts/libraries/BytesLib.sol#385-394) is never used and should be removed
BytesLib.toUint128(bytes,uint256) (contracts/libraries/BytesLib.sol#363-372) is never used and should be removed
BytesLib.toUint16(bytes,uint256) (contracts/libraries/BytesLib.sol#319-328) is never used and should be removed
BytesLib.toUint256(bytes,uint256) (contracts/libraries/BytesLib.sol#374-383) is never used and should be removed
BytesLib.toUint64(bytes,uint256) (contracts/libraries/BytesLib.sol#341-350) is never used and should be removed
BytesLib.toUint8(bytes,uint256) (contracts/libraries/BytesLib.sol#308-317) is never used and should be removed
BytesLib.toUint96(bytes,uint256) (contracts/libraries/BytesLib.sol#352-361) is never used and should be removed
ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
Flags.setFlag(uint256,uint256,bool) (contracts/libraries/Flags.sol#36-46) is never used and should be removed
SafeERC20Upgradeable.safeApprove(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#45-58) is never used and should be removed
SafeERC20Upgradeable.safeDecreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#69-80) is never used and should be removed
SafeERC20Upgradeable.safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#60-67) is never used and should be removed
SafeERC20Upgradeable.safeTransferFrom(IERC20Upgradeable,address,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#29-36) is never used and should be remov
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/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
Pragma version>=0.8.0<0.9.0 (contracts/libraries/BytesLib.sol#9) is too complex
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

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#58)
Low level call in AddressUpgradeable.functionCallWithValue(address,bytes,uint256,string) (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#132)
Low level call in AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#159)
Low level call in CallProxy.call(address,address,bytes,uint256,bytes,uint256) (contracts/periphery/CallProxy.sol#70-98):
        - (success) = _reserveAddress.call{value: amount}(new bytes(0)) (contracts/periphery/CallProxy.sol#95)
Reference: https://github.com/crytic/slither/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 AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 BytesLib.concat(bytes,bytes)._preBytes (contracts/libraries/BytesLib.sol#14) is not in mixedCase
Parameter BytesLib.concat(bytes,bytes)._postBytes (contracts/libraries/BytesLib.sol#15) is not in mixedCase
Parameter BytesLib.concatStorage(bytes,bytes)._preBytes (contracts/libraries/BytesLib.sol#91) is not in mixedCase
Parameter BytesLib.concatStorage(bytes,bytes)._postBytes (contracts/libraries/BytesLib.sol#91) is not in mixedCase
Parameter BytesLib.slice(bytes,uint256,uint256)._bytes (contracts/libraries/BytesLib.sol#229) is not in mixedCase
Parameter BytesLib.slice(bytes,uint256,uint256)._start (contracts/libraries/BytesLib.sol#230) is not in mixedCase
Parameter BytesLib.slice(bytes,uint256,uint256)._length (contracts/libraries/BytesLib.sol#231) is not in mixedCase
Parameter BytesLib.toAddress(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#297) is not in mixedCase
Parameter BytesLib.toAddress(bytes,uint256)._start (contracts/libraries/BytesLib.sol#297) is not in mixedCase
Parameter BytesLib.toUint8(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#308) is not in mixedCase
Parameter BytesLib.toUint8(bytes,uint256)._start (contracts/libraries/BytesLib.sol#308) is not in mixedCase
Parameter BytesLib.toUint16(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#319) is not in mixedCase
Parameter BytesLib.toUint16(bytes,uint256)._start (contracts/libraries/BytesLib.sol#319) is not in mixedCase
Parameter BytesLib.toUint32(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#330) is not in mixedCase
Parameter BytesLib.toUint32(bytes,uint256)._start (contracts/libraries/BytesLib.sol#330) is not in mixedCase
Parameter BytesLib.toUint64(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#341) is not in mixedCase
Parameter BytesLib.toUint64(bytes,uint256)._start (contracts/libraries/BytesLib.sol#341) is not in mixedCase
Parameter BytesLib.toUint96(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#352) is not in mixedCase
Parameter BytesLib.toUint96(bytes,uint256)._start (contracts/libraries/BytesLib.sol#352) is not in mixedCase
Parameter BytesLib.toUint128(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#363) is not in mixedCase
Parameter BytesLib.toUint128(bytes,uint256)._start (contracts/libraries/BytesLib.sol#363) is not in mixedCase
Parameter BytesLib.toUint256(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#374) is not in mixedCase
Parameter BytesLib.toUint256(bytes,uint256)._start (contracts/libraries/BytesLib.sol#374) is not in mixedCase
Parameter BytesLib.toBytes32(bytes,uint256)._bytes (contracts/libraries/BytesLib.sol#385) is not in mixedCase
Parameter BytesLib.toBytes32(bytes,uint256)._start (contracts/libraries/BytesLib.sol#385) is not in mixedCase
Parameter BytesLib.equal(bytes,bytes)._preBytes (contracts/libraries/BytesLib.sol#396) is not in mixedCase
Parameter BytesLib.equal(bytes,bytes)._postBytes (contracts/libraries/BytesLib.sol#396) is not in mixedCase
Parameter BytesLib.equalStorage(bytes,bytes)._preBytes (contracts/libraries/BytesLib.sol#440) is not in mixedCase
Parameter BytesLib.equalStorage(bytes,bytes)._postBytes (contracts/libraries/BytesLib.sol#441) is not in mixedCase
Parameter Flags.getFlag(uint256,uint256)._packedFlags (contracts/libraries/Flags.sol#25) is not in mixedCase
Parameter Flags.getFlag(uint256,uint256)._flag (contracts/libraries/Flags.sol#26) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._packedFlags (contracts/libraries/Flags.sol#37) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._flag (contracts/libraries/Flags.sol#38) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._value (contracts/libraries/Flags.sol#39) is not in mixedCase
Parameter CallProxy.call(address,address,bytes,uint256,bytes,uint256)._reserveAddress (contracts/periphery/CallProxy.sol#71) is not in mixedCase
Parameter CallProxy.call(address,address,bytes,uint256,bytes,uint256)._receiver (contracts/periphery/CallProxy.sol#72) is not in mixedCase
Parameter CallProxy.call(address,address,bytes,uint256,bytes,uint256)._data (contracts/periphery/CallProxy.sol#73) is not in mixedCase
Parameter CallProxy.call(address,address,bytes,uint256,bytes,uint256)._flags (contracts/periphery/CallProxy.sol#74) is not in mixedCase
Parameter CallProxy.call(address,address,bytes,uint256,bytes,uint256)._nativeSender (contracts/periphery/CallProxy.sol#75) is not in mixedCase
Parameter CallProxy.call(address,address,bytes,uint256,bytes,uint256)._chainIdFrom (contracts/periphery/CallProxy.sol#76) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._token (contracts/periphery/CallProxy.sol#102) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._reserveAddress (contracts/periphery/CallProxy.sol#103) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._receiver (contracts/periphery/CallProxy.sol#104) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._data (contracts/periphery/CallProxy.sol#105) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._flags (contracts/periphery/CallProxy.sol#106) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._nativeSender (contracts/periphery/CallProxy.sol#107) is not in mixedCase
Parameter CallProxy.callERC20(address,address,address,bytes,uint256,bytes,uint256)._chainIdFrom (contracts/periphery/CallProxy.sol#108) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

BytesLib.toAddress(bytes,uint256) (contracts/libraries/BytesLib.sol#297-306) uses literals with too many digits:
        - tempAddress = mload(uint256)(_bytes + 0x20 + _start) / 0x1000000000000000000000000 (contracts/libraries/BytesLib.sol#302)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is never used in CallProxy (contracts/periphery/CallProxy.sol#16-227)
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#170-174)
initialize() should be declared external:
        - CallProxy.initialize() (contracts/periphery/CallProxy.sol#63-65)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

DeBridgeToken.sol

```
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#31)
PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/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/utils/ContextUpgradeable.sol#31)
ERC20PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) shadows:
        - PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97)
        - ERC20Upgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#362)
        - ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,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

DeBridgeToken.initialize(string,string,uint8,address,address[]) (contracts/periphery/DeBridgeToken.sol#54-90) uses assembly
        - INLINE ASM (contracts/periphery/DeBridgeToken.sol#76-78)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

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/IAccessControlUpgradeable.sol#4)
        - ^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/token/ERC20/ERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - 0.8.7 (contracts/interfaces/IDeBridgeToken.sol#2)
        - 0.8.7 (contracts/interfaces/IERC20Permit.sol#3)
        - 0.8.7 (contracts/periphery/DeBridgeToken.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/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
ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
ERC20PausableUpgradeable.__ERC20Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#18-22) is never used and should be removed
ERC20PausableUpgradeable.__ERC20Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#24-25) is never used and should be removed
ERC20Upgradeable.__ERC20_init(string,string) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#55-58) is never used and should be removed
PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is never used and should be removed
PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is never used and should be removed
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/security/PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.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/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#51-55) is not in mixedCase
Function AccessControlUpgradeable.__AccessControl_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#57-58) is not in mixedCase
Variable AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable._Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is not in mixedCase
Function ERC20Upgradeable.__ERC20_init(string,string) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#55-58) is not in mixedCase
Function ERC20Upgradeable.__ERC20_init_unchained(string,string) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#60-63) is not in mixedCase
Variable ERC20Upgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol#362) is not in mixedCase
Function ERC20PausableUpgradeable.__ERC20Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#18-22) is not in mixedCase
Function ERC20PausableUpgradeable.__ERC20Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#24-25) is not in mixedCase
Variable ERC20PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 DeBridgeToken.mint(address,uint256)._receiver (contracts/periphery/DeBridgeToken.sol#93) is not in mixedCase
Parameter DeBridgeToken.mint(address,uint256)._amount (contracts/periphery/DeBridgeToken.sol#93) is not in mixedCase
Parameter DeBridgeToken.burn(uint256)._amount (contracts/periphery/DeBridgeToken.sol#98) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._owner (contracts/periphery/DeBridgeToken.sol#111) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._spender (contracts/periphery/DeBridgeToken.sol#112) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._value (contracts/periphery/DeBridgeToken.sol#113) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._deadline (contracts/periphery/DeBridgeToken.sol#114) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._v (contracts/periphery/DeBridgeToken.sol#115) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._r (contracts/periphery/DeBridgeToken.sol#116) is not in mixedCase
Parameter DeBridgeToken.permit(address,address,uint256,uint256,uint8,bytes32,bytes32)._s (contracts/periphery/DeBridgeToken.sol#117) is not in mixedCase
Variable DeBridgeToken.DOMAIN_SEPARATOR (contracts/periphery/DeBridgeToken.sol#21) is not in mixedCase
Variable DeBridgeToken._decimals (contracts/periphery/DeBridgeToken.sol#29) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

ERC20PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20PausableUpgradeable.sol#42) is never used in DeBridgeToken (contracts/periphery/DeBridgeToken.sol#10-166)
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#170-174)
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)
approve(address,uint256) should be declared external:
        - ERC20Upgradeable.approve(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)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

Claimer.sol

```
Reentrancy in DeBridgeGate._send(bytes,address,uint256,uint256,bool) (contracts/transfers/DeBridgeGate.sol#659-804):
        External calls:
        - _validateToken(_tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,None) = _token.call(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#726)
        - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/transfers/DeBridgeGate.sol#731)
        - _safeTransferETH(msg.sender,msg.value - nativeFee) (contracts/transfers/DeBridgeGate.sol#749)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        External calls sending eth:
        - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
        - _safeTransferETH(msg.sender,msg.value - nativeFee) (contracts/transfers/DeBridgeGate.sol#769)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        State variables written after the call(s):
        - debridge.balance += amountAfterFee (contracts/transfers/DeBridgeGate.sol#797)
        - debridge.balance -= amountAfterFee (contracts/transfers/DeBridgeGate.sol#808)
        - getDebridgeFeeInfo[nativeDebridgeId].collectedFees += 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#31)
PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) shadows:
        - ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variable-shadowing

Claimer.withdrawFee(address[]) (contracts/periphery/Claimer.sol#141-149) ignores return value by IERC20(_tokenAddresses[i]).transfer(msg.sender,IERC20(_tokenAddresses[i]).balanceOf(address(this))) (contracts/periphe
Claimer.withdrawSingleFee(address) (contracts/periphery/Claimer.sol#151-156) ignores return value by IERC20(_tokenAddresses).transfer(msg.sender,IERC20(_tokenAddresses).balanceOf(address(this))) (contracts/periphery
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer

DeBridgeGate._normalizeTokenAmount(address,uint256) (contracts/transfers/DeBridgeGate.sol#1011-1024) performs a multiplication on the result of a division:
        -_amount = _amount / multiplier * 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,uint32,bytes) (contracts/transfers/DeBridgeGate.sol#182-229) 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.deployNewAsset(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)
        - deBridgeTokenAddress = IDeBridgeTokenDeployer(deBridgeTokenDeployer).deployAsset(debridgeId,_name,_symbol,_decimals) (contracts/transfers/DeBridgeGate.sol#334-335)
        State variables written after the call(s):
        - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
                - debridge.exist = true (contracts/transfers/DeBridgeGate.sol#628)
                - debridge.tokenAddress = _tokenAddress (contracts/transfers/DeBridgeGate.sol#629)
                - debridge.chainId = _nativeChainId (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-1

DeBridgeGate.claim(bytes32,uint256,uint256,address,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,uint32,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.setDefiController(address) (contracts/transfers/DeBridgeGate.sol#462-464) should emit an event for:
        - defiController = _defiController (contracts/transfers/DeBridgeGate.sol#463)
DeBridgeGate.setFeeContractUpdater(address) (contracts/transfers/DeBridgeGate.sol#468-470) should emit an event for:
        - feeContractUpdater = _value (contracts/transfers/DeBridgeGate.sol#469)
DeBridgeGate.setFeeProxy(address) (contracts/transfers/DeBridgeGate.sol#547-549) should emit an event for:
        - feeProxy = _feeProxy (contracts/transfers/DeBridgeGate.sol#548)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

DeBridgeGate.initialize(uint8,IWETH) (contracts/transfers/DeBridgeGate.sol#168-177) should emit an event for:
        - excessConfirmations = _excessConfirmations (contracts/transfers/DeBridgeGate.sol#172)
DeBridgeGate.updateExcessConfirmations(uint8) (contracts/transfers/DeBridgeGate.sol#402-405) should emit an event for:
        - excessConfirmations = _excessConfirmations (contracts/transfers/DeBridgeGate.sol#404)
DeBridgeGate.updateFlashFee(uint256) (contracts/transfers/DeBridgeGate.sol#567-570) 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 :
                - callProxy = _callProxy (contracts/transfers/DeBridgeGate.sol#424)
DeBridgeGate.setSignatureVerifier(address)._verifier (contracts/transfers/DeBridgeGate.sol#450) lacks a zero-check on :
                - signatureVerifier = _verifier (contracts/transfers/DeBridgeGate.sol#451)
DeBridgeGate.setDeBridgeTokenDeployer(address)._deBridgeTokenDeployer (contracts/transfers/DeBridgeGate.sol#456) lacks a zero-check on :
                - deBridgeTokenDeployer = _deBridgeTokenDeployer (contracts/transfers/DeBridgeGate.sol#457)
DeBridgeGate.setDefiController(address)._defiController (contracts/transfers/DeBridgeGate.sol#462) lacks a zero-check on :
                - defiController = _defiController (contracts/transfers/DeBridgeGate.sol#463)
DeBridgeGate.setFeeContractUpdater(address)._value (contracts/transfers/DeBridgeGate.sol#468) lacks a zero-check on :
                - feeContractUpdater = _value (contracts/transfers/DeBridgeGate.sol#469)
DeBridgeGate.setFeeProxy(address)._feeProxy (contracts/transfers/DeBridgeGate.sol#547) lacks a zero-check on :
                - feeProxy = _feeProxy (contracts/transfers/DeBridgeGate.sol#548)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Claimer.batchClaim(Claimer.ClaimInfo[]) (contracts/periphery/Claimer.sol#66-85) has external calls inside a loop: deBridgeGate.claim(claim.debridgeId,claim.amount,claim.chainIdFrom,claim.receiver,claim.nonce,claim.
r.sol#72-83)
Claimer.batchAssetsDeploy(Claimer.AssetDeployInfo[]) (contracts/periphery/Claimer.sol#87-105) has external calls inside a loop: deBridgeGate.deployNewAsset(deploy.nativeTokenAddress,deploy.nativeChainId,deploy.name
acts/periphery/Claimer.sol#93-103)
Claimer.isSubmissionsUsed(bytes32[]) (contracts/periphery/Claimer.sol#110-118) has external calls inside a loop: result[i] = deBridgeGate.isSubmissionUsed(_submissionIds[i]) (contracts/periphery/Claimer.sol#116)
Claimer.isDebridgesExists(bytes32[]) (contracts/periphery/Claimer.sol#120-137) has external calls inside a loop: (exist) = deBridgeGate.getDebridge(_debridgeIds[i]) (contracts/periphery/Claimer.sol#126-134)
Claimer.withdrawFee(address[]) (contracts/periphery/Claimer.sol#141-149) has external calls inside a loop: IERC20(_tokenAddresses[i]).transfer(msg.sender,IERC20(_tokenAddresses[i]).balanceOf(address(this))) (contra
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#calls-inside-a-loop

Reentrancy in DeBridgeGate._send(bytes,address,uint256,uint256,bool) (contracts/transfers/DeBridgeGate.sol#659-804):
        External calls:
        - _validateToken(_tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,None) = _token.call(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)
        State variables written after the call(s):
        - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
                - getAmountThreshold[_debridgeId] = type()(uint256).max (contracts/transfers/DeBridgeGate.sol#638)
        - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
                - debridge.exist = true (contracts/transfers/DeBridgeGate.sol#628)
                - debridge.tokenAddress = _tokenAddress (contracts/transfers/DeBridgeGate.sol#629)
                - debridge.chainId = _nativeChainId (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)
        - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
                - tokenInfo.nativeChainId = _nativeChainId (contracts/transfers/DeBridgeGate.sol#642)
                - tokenInfo.nativeAddress = _nativeAddress (contracts/transfers/DeBridgeGate.sol#643)
Reentrancy in DeBridgeGate.deployNewAsset(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)
        - deBridgeTokenAddress = IDeBridgeTokenDeployer(deBridgeTokenDeployer).deployAsset(debridgeId,_name,_symbol,_decimals) (contracts/transfers/DeBridgeGate.sol#334-335)
        State variables written after the call(s):
        - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
                - getAmountThreshold[_debridgeId] = type()(uint256).max (contracts/transfers/DeBridgeGate.sol#638)
        - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
                - tokenInfo.nativeChainId = _nativeChainId (contracts/transfers/DeBridgeGate.sol#642)
                - tokenInfo.nativeAddress = _nativeAddress (contracts/transfers/DeBridgeGate.sol#643)
Reentrancy in DeBridgeGate.flash(address,address,uint256,bytes) (contracts/transfers/DeBridgeGate.sol#287-308):
        External calls:
        - IERC20Upgradeable(_tokenAddress).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#299)
        - IFlashCallback(msg.sender).flashCallback(currentFlashFee,_data) (contracts/transfers/DeBridgeGate.sol#300)
        State variables written after the call(s):
        - getDebridgeFeeInfo[debridgeId].collectedFees += paid (contracts/transfers/DeBridgeGate.sol#306)
Reentrancy in DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes) (contracts/transfers/DeBridgeGate.sol#182-229):
        External calls:
        - (amountAfterFee,debridgeId,feeParams) = _send{_permit:_tokenAddress,_amount:_chainIdTo,_useAssetFee} (contracts/transfers/DeBridgeGate.sol#196-202)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - IERC20Permit(_tokenAddress).permit(msg.sender,address(this),_amount,deadline,v,r,s) (contracts/transfers/DeBridgeGate.sol#677-684)
                - (success,None) = _token.call(abi.encodeWithSignature(symbol())) (contracts/transfers/DeBridgeGate.sol#889)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
                - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/transfers/DeBridgeGate.sol#731)
                - IDeBridgeToken(debridge.tokenAddress).burn(amountAfterFee) (contracts/transfers/DeBridgeGate.sol#811)
```

```
        External calls sending eth:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
              - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
        State variables written after the call(s):
        - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
              - nonce ++ (contracts/transfers/DeBridgeGate.sol#868)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in DeBridgeGate._claim(bytes32,bytes32,address,uint256,uint256,IDeBridgeGate.SubmissionAutoParamsFrom) (contracts/transfers/DeBridgeGate.sol#898-966):
        External calls:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
              - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
              - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - status = ICallProxy(_callProxy).call(_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#931-938)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
              - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - status = ICallProxy(_callProxy).callERC20(_token,_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#943-9
        External calls sending eth:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        Event emitted after the call(s):
        - AutoRequestExecuted(_submissionId,status,_callProxy) (contracts/transfers/DeBridgeGate.sol#953)
Reentrancy in DeBridgeGate._claim(bytes32,bytes32,address,uint256,uint256,IDeBridgeGate.SubmissionAutoParamsFrom) (contracts/transfers/DeBridgeGate.sol#898-966):
        External calls:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
              - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
              - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - status = ICallProxy(_callProxy).call(_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#931-938)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
              - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - status = ICallProxy(_callProxy).callERC20(_token,_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#943-9
        - _withdrawWeth(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#956)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
              - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _mintOrTransfer(_token,_receiver,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#958)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
              - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        External calls sending eth:
        - _mintOrTransfer(_token,msg.sender,_autoParams.executionFee,isNativeToken) (contracts/transfers/DeBridgeGate.sol#922)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_callProxy,_amount) (contracts/transfers/DeBridgeGate.sol#930)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _mintOrTransfer(_token,_callProxy,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#941)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _withdrawWeth(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#956)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        - _mintOrTransfer(_token,_receiver,_amount,isNativeToken) (contracts/transfers/DeBridgeGate.sol#958)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        Event emitted after the call(s):
        - MonitoringClaimEvent(_submissionId,debridge.balance,IERC20Upgradeable(debridge.tokenAddress).totalSupply()) (contracts/transfers/DeBridgeGate.sol#961-965)
Reentrancy in DeBridgeGate._send(bytes,address,uint256,uint256,bool) (contracts/transfers/DeBridgeGate.sol#659-804):
        External calls:
        - _validateToken(_tokenAddress) (contracts/transfers/DeBridgeGate.sol#670)
              - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
              - (success,None) = _token.call(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)
        Event emitted after the call(s):
        - PairAdded(_debridgeId,_tokenAddress,_nativeAddress,_nativeChainId,debridge.maxAmount,debridge.minReservesBps) (contracts/transfers/DeBridgeGate.sol#645-652)
              - _addAsset(debridgeId,assetAddress,abi.encodePacked(assetAddress),getChainId()) (contracts/transfers/DeBridgeGate.sol#711-716)
Reentrancy in Claimer.batchAssetsDeploy(Claimer.AssetDeployInfo[]) (contracts/periphery/Claimer.sol#87-105):
        External calls:
        - deBridgeGate.deployNewAsset(deploy.nativeTokenAddress,deploy.nativeChainId,deploy.name,deploy.symbol,deploy.decimals,deploy.signatures) (contracts/periphery/Claimer.sol#93-103)
        Event emitted after the call(s):
        - BatchError(i) (contracts/periphery/Claimer.sol#102)
Reentrancy in Claimer.batchClaim(Claimer.ClaimInfo[]) (contracts/periphery/Claimer.sol#66-85):
        External calls:
        - deBridgeGate.claim(claim.debridgeId,claim.amount,claim.chainIdFrom,claim.receiver,claim.nonce,claim.signatures,claim.autoParams) (contracts/periphery/Claimer.sol#72-83)
        Event emitted after the call(s):
        - BatchError(i) (contracts/periphery/Claimer.sol#82)
Reentrancy in DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes) (contracts/transfers/DeBridgeGate.sol#232-284):
        External calls:
        - _checkConfirmations(submissionId,_debridgeId,_amount,_signatures) (contracts/transfers/DeBridgeGate.sol#263)
              - ISignatureVerifier(signatureVerifier).submit(_submissionId,_signatures,excessConfirmations) (contracts/transfers/DeBridgeGate.sol#605-609)
              - ISignatureVerifier(signatureVerifier).submit(_submissionId,_signatures,0) (contracts/transfers/DeBridgeGate.sol#605-609)
        - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
              - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
              - weth.withdraw(_amount) (contracts/transfers/DeBridgeGate.sol#997)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - IERC20Upgradeable(_token).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#976)
              - IDeBridgeToken(_token).mint(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#978)
              - IERC20Upgradeable(address(weth)).safeTransfer(address(wethGate),_amount) (contracts/transfers/DeBridgeGate.sol#1001)
              - wethGate.withdraw(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#1002)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
              - status = ICallProxy(_callProxy).call(_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.sol#931-938)
              - status = ICallProxy(_callProxy).callERC20(_token,_autoParams.fallbackAddress,_receiver,_autoParams.data,_autoParams.flags,_autoParams.nativeSender,_chainIdFrom) (contracts/transfers/DeBridgeGate.s
        External calls sending eth:
        - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
              - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
              - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
        Event emitted after the call(s):
        - AutoRequestExecuted(_submissionId,status,_callProxy) (contracts/transfers/DeBridgeGate.sol#953)
              - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
        - Claimed(submissionId,_debridgeId,_amount,_receiver,_nonce,_chainIdFrom,_autoParams,isNativeToken) (contracts/transfers/DeBridgeGate.sol#274-283)
        - MonitoringClaimEvent(_submissionId,debridge.balance,IERC20Upgradeable(debridge.tokenAddress).totalSupply()) (contracts/transfers/DeBridgeGate.sol#961-965)
              - isNativeToken = _claim(submissionId,_debridgeId,_receiver,_amount,_chainIdFrom,autoParams) (contracts/transfers/DeBridgeGate.sol#265-272)
Reentrancy in DeBridgeGate.deployNewAsset(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)
        - deBridgeTokenAddress = IDeBridgeTokenDeployer(deBridgeTokenDeployer).deployAsset(debridgeId,_name,_symbol,_decimals) (contracts/transfers/DeBridgeGate.sol#334-335)
        Event emitted after the call(s):
        - PairAdded(_debridgeId,_tokenAddress,_nativeAddress,_nativeChainId,debridge.maxAmount,debridge.minReservesBps) (contracts/transfers/DeBridgeGate.sol#645-652)
              - _addAsset(debridgeId,deBridgeTokenAddress,_nativeTokenAddress,_nativeChainId) (contracts/transfers/DeBridgeGate.sol#337)
Reentrancy in DeBridgeGate.flash(address,address,uint256,bytes) (contracts/transfers/DeBridgeGate.sol#287-308):
        External calls:
        - IERC20Upgradeable(_tokenAddress).safeTransfer(_receiver,_amount) (contracts/transfers/DeBridgeGate.sol#299)
```

```
        - IFlashCallback(msg.sender).flashCallback(currentFlashFee,_data) (contracts/transfers/DeBridgeGate.sol#380)
        Event emitted after the call(s):
        - Flash(msg.sender,_tokenAddress,_receiver,_amount,paid) (contracts/transfers/DeBridgeGate.sol#307)
Reentrancy in DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes) (contracts/transfers/DeBridgeGate.sol#182-229):
        External calls:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#93)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - IERC20Permit(_tokenAddress).permit(msg.sender,address(this),_amount,deadline,v,r,s) (contracts/transfers/DeBridgeGate.sol#677-684)
                - (success,None) = _token.call(abi.encodeWithSignature(symbol())) (contracts/transfers/DeBridgeGate.sol#889)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
                - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/transfers/DeBridgeGate.sol#731)
                - IDeBridgeToken(debridge.tokenAddress).burn(amountAfterFee) (contracts/transfers/DeBridgeGate.sol#801)
        External calls sending eth:
        - (amountAfterFee,debridgeId,feeParams) = _send(_permit,_tokenAddress,_amount,_chainIdTo,_useAssetFee) (contracts/transfers/DeBridgeGate.sol#196-202)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
                - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#132)
                - weth.deposit{value: _amount}() (contracts/transfers/DeBridgeGate.sol#726)
        Event emitted after the call(s):
        - MonitoringSendEvent(submissionId,nonce,getDebridge[_debridgeId].balance,IERC20Upgradeable(getDebridge[_debridgeId].tokenAddress).totalSupply()) (contracts/transfers/DeBridgeGate.sol#860-865)
                - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
        - Sent(submissionId,_debridgeId,_amount,_receiver,nonce,_chainIdTo,_referralCode,feeParams,abi.encode(autoParams),msg.sender) (contracts/transfers/DeBridgeGate.sol#846-857)
                - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
        - Sent(submissionId,_debridgeId,_amount,_receiver,nonce,_chainIdTo,_referralCode,feeParams,bytes(),msg.sender) (contracts/transfers/DeBridgeGate.sol#846-857)
                - _publishSubmission(debridgeId,_chainIdTo,amountAfterFee,_receiver,feeParams,_referralCode,autoParams,_autoParams.length > 0) (contracts/transfers/DeBridgeGate.sol#219-228)
Reentrancy in DeBridgeGate.withdrawFee(bytes32) (contracts/transfers/DeBridgeGate.sol#489-506):
        External calls:
        - _safeTransferETH(feeProxy,amount) (contracts/transfers/DeBridgeGate.sol#499)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        - IERC20Upgradeable(getDebridge[_debridgeId].tokenAddress).safeTransfer(feeProxy,amount) (contracts/transfers/DeBridgeGate.sol#503)
        External calls sending eth:
        - _safeTransferETH(feeProxy,amount) (contracts/transfers/DeBridgeGate.sol#499)
                - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
        Event emitted after the call(s):
        - WithdrawnFee(_debridgeId,amount) (contracts/transfers/DeBridgeGate.sol#505)
Reference: https://github.com/crytic/slither/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.verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#169-189) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#181-184)
SignatureUtil.parseSignature(bytes,uint256) (contracts/libraries/SignatureUtil.sol#32-49) uses assembly
        - INLINE ASM (contracts/libraries/SignatureUtil.sol#41-45)
SignatureUtil.toUint256(bytes,uint256) (contracts/libraries/SignatureUtil.sol#51-61) uses assembly
        - INLINE ASM (contracts/libraries/SignatureUtil.sol#58-60)
DeBridgeGate.getChainId() (contracts/transfers/DeBridgeGate.sol#1134-1138) uses assembly
        - INLINE ASM (contracts/transfers/DeBridgeGate.sol#1135-1137)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

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/IAccessControlUpgradeable.sol#4)
        - ^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/security/ReentrancyGuardUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/StringsUpgradeable.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/IERC165Upgradeable.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4)
        - 0.8.7 (contracts/interfaces/ICallProxy.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeGate.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeToken.sol#2)
        - 0.8.7 (contracts/interfaces/IDeBridgeTokenDeployer.sol#2)
        - 0.8.7 (contracts/interfaces/IERC20Permit.sol#3)
        - 0.8.7 (contracts/interfaces/IFlashCallback.sol#2)
        - 0.8.7 (contracts/interfaces/ISignatureVerifier.sol#2)
        - 0.8.7 (contracts/interfaces/IWETH.sol#2)
        - 0.8.7 (contracts/interfaces/IWethGate.sol#2)
        - 0.8.7 (contracts/libraries/Flags.sol#2)
        - 0.8.7 (contracts/libraries/SignatureUtil.sol#2)
        - 0.8.7 (contracts/periphery/Claimer.sol#2)
        - 0.8.7 (contracts/transfers/DeBridgeGate.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/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.functionCall(address,bytes) (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#109-115) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#142-144) is never used and should be removed
AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161) 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
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#21-23) is never used and should be removed
ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) 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._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#24-26) is never used and should be removed
ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#28-29) is never used and should be removed
ERC20._burn(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#275-290) is never used and should be removed
ERC20._mint(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#252-262) is never used and should be removed
Flags.setFlag(uint256,uint256,bool) (contracts/libraries/Flags.sol#36-45) is never used and should be removed
PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is never used and should be removed
PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is never used and should be removed
SafeERC20Upgradeable.safeApprove(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#45-58) is never used and should be removed
SafeERC20Upgradeable.safeDecreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#69-80) is never used and should be removed
SafeERC20Upgradeable.safeIncreaseAllowance(IERC20Upgradeable,address,uint256) (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol#60-67) is never used and should be removed
SignatureUtil.getUnsignedMsg(bytes32) (contracts/libraries/SignatureUtil.sol#13-15) is never used and should be removed
SignatureUtil.splitSignature(bytes) (contracts/libraries/SignatureUtil.sol#19-30) is never used and should be removed
StringsUpgradeable.toHexString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol#40-51) is never used and should be removed
StringsUpgradeable.toString(uint256) (node_modules/@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.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/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/security/PausableUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/extensions/IERC20MetadataUpgradeable.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.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/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
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol#4) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#4) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-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#58)
Low level call in AddressUpgradeable.functionCallWithValue(address,bytes,uint256,string) (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#132)
Low level call in AddressUpgradeable.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#152-161):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol#159)
Low level call in DeBridgeGate._validateToken(address) (contracts/transfers/DeBridgeGate.sol#878-891):
        - (success) = _token.call(abi.encodeWithSignature(decimals())) (contracts/transfers/DeBridgeGate.sol#885)
        - (success,None) = _token.call(abi.encodeWithSignature(symbol())) (contracts/transfers/DeBridgeGate.sol#889)
Low level call in DeBridgeGate._safeTransferETH(address,uint256) (contracts/transfers/DeBridgeGate.sol#988-991):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/transfers/DeBridgeGate.sol#989)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

AUTOMATED TESTING

```
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 AccessControlUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/access/AccessControlUpgradeable.sol#232) is not in mixedCase
Function PausableUpgradeable.__Pausable_init() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#34-37) is not in mixedCase
Function PausableUpgradeable.__Pausable_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#39-41) is not in mixedCase
Variable PausableUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol#97) is not in mixedCase
Function ReentrancyGuardUpgradeable.__ReentrancyGuard_init() (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#40-42) is not in mixedCase
Function ReentrancyGuardUpgradeable.__ReentrancyGuard_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#44-46) is not in mixedCase
Variable ReentrancyGuardUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#68) is not in mixedCase
Function ContextUpgradeable.__Context_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#18-20) is not in mixedCase
Function ContextUpgradeable.__Context_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#22-23) is not in mixedCase
Variable ContextUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol#31) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init() (node_modules/@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol#24-26) is not in mixedCase
Function ERC165Upgradeable.__ERC165_init_unchained() (node_modules/@openzeppelin/contracts-upgradeable/utils/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 Flags.getFlag(uint256,uint256)._packedFlags (contracts/libries/Flags.sol#25) is not in mixedCase
Parameter Flags.getFlag(uint256,uint256)._flag (contracts/libraries/Flags.sol#26) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._packedFlags (contracts/libraries/Flags.sol#37) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._flag (contracts/libraries/Flags.sol#38) is not in mixedCase
Parameter Flags.setFlag(uint256,uint256,bool)._value (contracts/libraries/Flags.sol#39) 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#19) is not in mixedCase
Parameter SignatureUtil.parseSignature(bytes,uint256)._signatures (contracts/libraries/SignatureUtil.sol#32) is not in mixedCase
Parameter SignatureUtil.toUint256(bytes,uint256)._bytes (contracts/libraries/SignatureUtil.sol#61) is not in mixedCase
Parameter SignatureUtil.toUint256(bytes,uint256)._offset (contracts/libraries/SignatureUtil.sol#51) is not in mixedCase
Parameter Claimer.initialize(DeBridgeGate)._deBridgeGate (contracts/periphery/Claimer.sol#59) is not in mixedCase
Parameter Claimer.batchClaim(Claimer.ClaimInfo[])._claims (contracts/periphery/Claimer.sol#67) is not in mixedCase
Parameter Claimer.batchAssetsDeploy(Claimer.AssetDeployInfo[])._deploys (contracts/periphery/Claimer.sol#88) is not in mixedCase
Parameter Claimer.isSubmissionsUsed(bytes32[])._submissionIds (contracts/periphery/Claimer.sol#111) is not in mixedCase
Parameter Claimer.isDebridgesExists(bytes32[])._debridgeIds (contracts/periphery/Claimer.sol#121) is not in mixedCase
Parameter Claimer.withdrawFee(address[])._tokenAddresses (contracts/periphery/Claimer.sol#142) is not in mixedCase
Parameter Claimer.withdrawSingleFee(address)._tokenAddress (contracts/periphery/Claimer.sol#151) is not in mixedCase
Parameter Claimer.setDeBridgeGate(DeBridgeGate)._deBridgeGate (contracts/periphery/Claimer.sol#158) is not in mixedCase
Parameter DeBridgeGate.initialize(uint8,IWETH)._excessConfirmations (contracts/transfers/DeBridgeGate.sol#169) is not in mixedCase
Parameter DeBridgeGate.initialize(uint8,IWETH)._weth (contracts/transfers/DeBridgeGate.sol#170) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._tokenAddress (contracts/transfers/DeBridgeGate.sol#183) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._amount (contracts/transfers/DeBridgeGate.sol#184) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._chainIdTo (contracts/transfers/DeBridgeGate.sol#185) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._receiver (contracts/transfers/DeBridgeGate.sol#186) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._permit (contracts/transfers/DeBridgeGate.sol#187) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._useAssetFee (contracts/transfers/DeBridgeGate.sol#188) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._referralCode (contracts/transfers/DeBridgeGate.sol#189) is not in mixedCase
Parameter DeBridgeGate.send(address,uint256,uint256,bytes,bytes,bool,uint32,bytes)._autoParams (contracts/transfers/DeBridgeGate.sol#190) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._debridgeId (contracts/transfers/DeBridgeGate.sol#233) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._amount (contracts/transfers/DeBridgeGate.sol#234) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._chainIdFrom (contracts/transfers/DeBridgeGate.sol#235) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._receiver (contracts/transfers/DeBridgeGate.sol#236) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._nonce (contracts/transfers/DeBridgeGate.sol#237) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._signatures (contracts/transfers/DeBridgeGate.sol#238) is not in mixedCase
Parameter DeBridgeGate.claim(bytes32,uint256,uint256,address,uint256,bytes,bytes)._autoParams (contracts/transfers/DeBridgeGate.sol#239) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._tokenAddress (contracts/transfers/DeBridgeGate.sol#288) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._receiver (contracts/transfers/DeBridgeGate.sol#289) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._amount (contracts/transfers/DeBridgeGate.sol#290) is not in mixedCase
Parameter DeBridgeGate.flash(address,address,uint256,bytes)._data (contracts/transfers/DeBridgeGate.sol#291) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._nativeTokenAddress (contracts/transfers/DeBridgeGate.sol#318) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._nativeChainId (contracts/transfers/DeBridgeGate.sol#319) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._name (contracts/transfers/DeBridgeGate.sol#320) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._symbol (contracts/transfers/DeBridgeGate.sol#321) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._decimals (contracts/transfers/DeBridgeGate.sol#322) is not in mixedCase
Parameter DeBridgeGate.deployNewAsset(bytes,uint256,string,string,uint8,bytes)._signatures (contracts/transfers/DeBridgeGate.sol#323) is not in mixedCase
Parameter DeBridgeGate.autoUpdateFixedNativeFee(uint256)._globalFixedNativeFee (contracts/transfers/DeBridgeGate.sol#343) is not in mixedCase
Parameter DeBridgeGate.updateChainSupport(uint256[],IDeBridgeGate.ChainSupportInfo[],bool)._chainIds (contracts/transfers/DeBridgeGate.sol#356) is not in mixedCase
Parameter DeBridgeGate.updateChainSupport(uint256[],IDeBridgeGate.ChainSupportInfo[],bool)._chainSupportInfo (contracts/transfers/DeBridgeGate.sol#357) is not in mixedCase
Parameter DeBridgeGate.updateChainSupport(uint256[],IDeBridgeGate.ChainSupportInfo[],bool)._isChainFrom (contracts/transfers/DeBridgeGate.sol#358) is not in mixedCase
Parameter DeBridgeGate.updateGlobalFee(uint256,uint16)._globalFixedNativeFee (contracts/transfers/DeBridgeGate.sol#376) is not in mixedCase
Parameter DeBridgeGate.updateGlobalFee(uint256,uint16)._globalTransferFeeBps (contracts/transfers/DeBridgeGate.sol#377) is not in mixedCase
Parameter DeBridgeGate.updateAssetFixedFees(bytes32,uint256[],uint256[])._debridgeId (contracts/transfers/DeBridgeGate.sol#389) is not in mixedCase
Parameter DeBridgeGate.updateAssetFixedFees(bytes32,uint256[],uint256[])._supportedChainIds (contracts/transfers/DeBridgeGate.sol#390) is not in mixedCase
Parameter DeBridgeGate.updateAssetFixedFees(bytes32,uint256[],uint256[])._assetFeesInfo (contracts/transfers/DeBridgeGate.sol#391) is not in mixedCase
Parameter DeBridgeGate.updateExcessConfirmations(uint8)._excessConfirmations (contracts/transfers/DeBridgeGate.sol#402) is not in mixedCase
Parameter DeBridgeGate.setChainSupport(uint256,bool,bool)._chainId (contracts/transfers/DeBridgeGate.sol#411) is not in mixedCase
Parameter DeBridgeGate.setChainSupport(uint256,bool,bool)._isSupported (contracts/transfers/DeBridgeGate.sol#411) is not in mixedCase
Parameter DeBridgeGate.setChainSupport(uint256,bool,bool)._isChainFrom (contracts/transfers/DeBridgeGate.sol#411) is not in mixedCase
Parameter DeBridgeGate.setCallProxy(address)._callProxy (contracts/transfers/DeBridgeGate.sol#423) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._debridgeId (contracts/transfers/DeBridgeGate.sol#434) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._maxAmount (contracts/transfers/DeBridgeGate.sol#435) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._minReservesBps (contracts/transfers/DeBridgeGate.sol#436) is not in mixedCase
Parameter DeBridgeGate.updateAsset(bytes32,uint256,uint16,uint256)._amountThreshold (contracts/transfers/DeBridgeGate.sol#437) is not in mixedCase
Parameter DeBridgeGate.setSignatureVerifier(address)._verifier (contracts/transfers/DeBridgeGate.sol#450) is not in mixedCase
Parameter DeBridgeGate.setDeBridgeTokenDeployer(address)._deBridgeTokenDeployer (contracts/transfers/DeBridgeGate.sol#456) is not in mixedCase
Parameter DeBridgeGate.setDefiController(address)._defiController (contracts/transfers/DeBridgeGate.sol#462) is not in mixedCase
Parameter DeBridgeGate.setFeeContractUpdater(address)._value (contracts/transfers/DeBridgeGate.sol#468) is not in mixedCase
Parameter DeBridgeGate.setWethGate(IWethGate)._wethGate (contracts/transfers/DeBridgeGate.sol#474) is not in mixedCase
Parameter DeBridgeGate.withdrawFee(bytes32)._debridgeId (contracts/transfers/DeBridgeGate.sol#489) is not in mixedCase
Parameter DeBridgeGate.requestReserves(address,uint256)._tokenAddress (contracts/transfers/DeBridgeGate.sol#509) is not in mixedCase
Parameter DeBridgeGate.requestReserves(address,uint256)._amount (contracts/transfers/DeBridgeGate.sol#509) is not in mixedCase
Parameter DeBridgeGate.returnReserves(address,uint256)._tokenAddress (contracts/transfers/DeBridgeGate.sol#528) is not in mixedCase
Parameter DeBridgeGate.returnReserves(address,uint256)._amount (contracts/transfers/DeBridgeGate.sol#528) is not in mixedCase
Parameter DeBridgeGate.setFeeProxy(address)._feeProxy (contracts/transfers/DeBridgeGate.sol#547) is not in mixedCase
Parameter DeBridgeGate.blockSubmission(bytes32[],bool)._submissionIds (contracts/transfers/DeBridgeGate.sol#556) is not in mixedCase
Parameter DeBridgeGate.updateFlashFee(uint256)._flashFeeBps (contracts/transfers/DeBridgeGate.sol#567) is not in mixedCase
Parameter DeBridgeGate.updateFeeDiscount(address,uint16,uint16)._address (contracts/transfers/DeBridgeGate.sol#577) is not in mixedCase
Parameter DeBridgeGate.updateFeeDiscount(address,uint16,uint16)._discountFixBps (contracts/transfers/DeBridgeGate.sol#578) is not in mixedCase
Parameter DeBridgeGate.updateFeeDiscount(address,uint16,uint16)._discountTransferBps (contracts/transfers/DeBridgeGate.sol#579) is not in mixedCase
Parameter DeBridgeGate.getDefiAvailableReserves(address)._tokenAddress (contracts/transfers/DeBridgeGate.sol#1029) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,address)._chainId (contracts/transfers/DeBridgeGate.sol#1042) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,address)._tokenAddress (contracts/transfers/DeBridgeGate.sol#1042) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,bytes)._chainId (contracts/transfers/DeBridgeGate.sol#1049) is not in mixedCase
Parameter DeBridgeGate.getDebridgeId(uint256,bytes)._tokenAddress (contracts/transfers/DeBridgeGate.sol#1049) is not in mixedCase
Parameter DeBridgeGate.getDebridgeChainAssetFixedFee(bytes32,uint256)._debridgeId (contracts/transfers/DeBridgeGate.sol#1055) is not in mixedCase
Parameter DeBridgeGate.getDebridgeChainAssetFixedFee(bytes32,uint256)._chainId (contracts/transfers/DeBridgeGate.sol#1056) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._debridgeId (contracts/transfers/DeBridgeGate.sol#1072) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._chainIdFrom (contracts/transfers/DeBridgeGate.sol#1073) is not in mixedCase
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)._autoParams (contracts/transfers/DeBridgeGate.sol#1077) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._hasAutoParams (contracts/transfers/DeBridgeGate.sol#1078) is not in mixedCase
Parameter DeBridgeGate.getSubmissionIdFrom(bytes32,uint256,uint256,address,uint256,IDeBridgeGate.SubmissionAutoParamsFrom,bool,address)._sender (contracts/transfers/DeBridgeGate.sol#1079) 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)._symbol (contracts/transfers/DeBridgeGate.sol#1122) is not in mixedCase
Parameter DeBridgeGate.getDeployId(bytes32,string,string,uint8)._decimals (contracts/transfers/DeBridgeGate.sol#1123) 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#232) is never used in Claimer (contracts/periphery/Claimer.sol#9-166)
ReentrancyGuardUpgradeable.__gap (node_modules/@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol#68) is never used in DeBridgeGate (contracts/transfers/DeBridgeGate.sol#26-1145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DeBridgeGate.lockedClaim (contracts/transfers/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/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#170-174)
name() should be declared external:
        - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#62-64)
symbol() should be declared external:
        - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#70-72)
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(DeBridgeGate) should be declared external:
        - Claimer.initialize(DeBridgeGate) (contracts/periphery/Claimer.sol#58-64)
initialize(uint8,IWETH) should be declared external:
        - DeBridgeGate.initialize(uint8,IWETH) (contracts/transfers/DeBridgeGate.sol#168-177)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

MultiSendCallOnly.sol

```
MultiSendCallOnly._multiSend(bytes) (contracts/libraries/MultiSendCallOnly.sol#21-60) uses assembly
        - INLINE ASM (contracts/libraries/MultiSendCallOnly.sol#23-59)
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-120) Weak Sources of Randomness from Chain Attributes	Low	Potential use of "block.number" as source of randomness.
103	(SWC-120) 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-113) 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.
83	(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 block.timestamp 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