MythX

REPORT 5EFDCC1644AD2C0019F802E1

Created	Thu Jul 02 2020 11:59:18 GMT+0000 (Coordinated Universal Time)
Number of analyses	3
User	goncalo05@gmail.com

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
092dda5e-f460-494f-b6cb-876347596d8b	src/Loihi.sol	47
3015c108-9341-4df0-8e94-208cf4067159	src/LoihiRoot.sol	24
9869b2a2-bb37-4eb5-b07d-68f1090cf2c4	src/ShellsExternal.sol	1

Analysis 092dda5e-f460-494f-b6cb-876347596d8b MythX

Started	Thu Jul 02 2020 12:15:20 GMT+0000 (Coordinated Universal Time)
Finished	Thu Jul 02 2020 13:00:39 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Mythx-Cli-0.6.19
Main Source File	Src/Loihi.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
14	12	21

ISSUES

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file
src/Loihi.sol
Locations

```
415 if (oBals_[i] == 0 && nBals_[i] == 0) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
416
417 oGLiq_ += oBals_[i];
418 nGLiq_ += nBals_[i];
```

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file
src/Loihi.sol
Locations

```
416
417 oGLiq_ += oBals_[i];
418 nGLiq_ += nBals_[i];
419
420 }
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

src/Shells.sol

Locations

```
251
252 if (_oBals[i] > _oHalt) revert("Shell/lower-halt");
253 if (_nHalt - _nBals[i] > _oHalt - _oBals[i]) revert("Shel/lower-halt");
254
255 }
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

src/Shells.sol

Locations

```
251
252 if (_oBals[i] > _oHalt) revert("Shell/lower-halt");
253 if (_nHalt - _nBals[i] > _oHalt - _oBals[i]) revert("Shel/lower-halt");
254
255 }
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

src/Shells.sol

Locations

```
95 if (_bal < _threshold) {
96
97 int128 _feeSection = _threshold - _bal;
98
99 fee_ = _feeSection.unsafe_div(_ideal);
```

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file

lib/abdk-libraries-solidity/src/ABDKMath64x64.sol

Locations

```
172 */
173 function unsafe_mul (int128 x, int128 y) internal pure returns (int128) {
174 int256 result = int256(x) * y >> 64;
175 return int128 (result);
176 }
```

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file

src/Loihi.sol

Locations

```
368 if (oBals_[i] == 0 && nBals_[i] == 0) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
369
370 oGLiq_ += oBals_[i];
371 nGLiq_ += nBals_[i];
```

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file

src/Loihi.sol

Locations

```
369
370 oGLiq_ += oBals_[i];
371 nGLiq_ += nBals_[i];
372
373 }
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

src/Shells.sol

Locations

```
236
237 if (_oBals[i] < _oHalt) revert("Shell/upper-halt");
238 if (_nBals[i] - _nHalt > _oBals[i] - _oHalt) revert("Shell/upper-halt");
239
240 }
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

src/Shells.sol

Locations

```
236
237 if (_oBals[i] < _oHalt) revert("Shell/upper-halt");
238 if (_nBals[i] - _nHalt > _oBals[i] - _oHalt) revert("Shell/upper-halt");
239
240 }
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

src/Shells.sol

Locations

```
112 if (_bal > _threshold) {
113
114 int128 _feeSection = _bal - _threshold;
115
116 fee_ = _feeSection.unsafe_div(_ideal);
```

HIGH

SWC-101

The arithmetic operation can underflow.

It is possible to cause an arithmetic underflow. Prevent the underflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the underflow.

Source file

lib/abdk-libraries-solidity/src/ABDKMath64x64.sol

Locations

```
134 */
135 function sub (int128 x, int128 y) internal pure returns (int128) {
136 int256 result = int256(x) - y;
137 require (result >= MIN_64x64 && result <= MAX_64x64);
138 return int128 (result);
```

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file

src/Shells.sol

Locations

```
77 for (uint i = 0; i < _weights.length; i++) {
78 int128 _ideal = _gLiq.unsafe_mul(_weights[i]);
79 psi_ += calculateMicroFee(_bals[i], _ideal, _beta, _delta);
80 }
```

HIGH

SWC-101

The arithmetic operation can overflow.

It is possible to cause an arithmetic overflow. Prevent the overflow by constraining inputs using the require() statement or use the OpenZeppelin SafeMath library for integer arithmetic operations. Refer to the transaction trace generated for this issue to reproduce the overflow.

Source file

lib/abdk-libraries-solidity/src/ABDKMath64x64.sol

Locations

```
159 */
160 function mul (int128 x, int128 y) internal pure returns (int128) {
161 int256 result = int256(x) * y >> 64;
162 require (result >= MIN_64x64 && result <= MAX_64x64);
163 return int128 (result);
```

MEDIUM

Incorrect ERC20 implementation

Contract "Loihi" looks like its trying to implement the ERC20 standard, but its missing a required event with signature "event Transfer(address indexed, address indexed, uint256)"

SWC-000

Source file

`src/Loihi.sol`

Locations

```
28  }
29
30  contract Loihi is LoihiRoot {
31
32  using ABDKMath64x64 for int128;
33  using ABDKMath64x64 for uint256;
34
35  using Assimilators for address;
36  using Shells for Shells.Shell;
37  using ShellsExternal for Shells.Shell;
38  using Controller for Shells.Shell;
39
40  event ShellsMinted(address indexed minter, uint256 amount, address[] indexed coins, uint256[] amounts);
41  event ShellsBurned(address indexed burner, uint256 amount, address[] indexed coins, uint256[] amounts);
42  event Trade(address indexed trader, address indexed origin, address indexed target, uint256 originAmount, uint256 targetAmount);
43  event SetFrozen(bool isFrozen);
44
45  // constructor () public {
46  constructor () public {
47
48  owner = msg.sender;
49  emit OwnershipTransferred(address(0), msg.sender);
50
51  shell.testHalts = true;
52
53  }
54
55  function setParams (uint256 _alpha, uint256 _beta, uint256 _epsilon, uint256 _max, uint256 _lambda) public onlyOwner {
56  maxFee = shell.setParams(_alpha, _beta, _epsilon, _max, _lambda);
57  }
58
59  function includeAsset (address _numeraire, address _nAssim, address _reserve, address _rAssim, uint256 _weight) public onlyOwner {
60  shell.includeAsset(_numeraire, _nAssim, _reserve, _rAssim, _weight);
61  }
62
63  function includeAssimilator (address _numeraire, address _derivative, address _assimilator) public onlyOwner {
64  shell.includeAssimilator(_numeraire, _derivative, _assimilator);
65  }
66
67  function excludeAdapter (address _assimilator) external onlyOwner {
68  delete shell.assimilators[_assimilator];
69  }
70
71  function supportsInterface (bytes4 interfaceID) public returns (bool) {
72  return interfaceID == ERC20ID || interfaceID == ERC165ID;
73  }
74
75  function freeze (bool _isFrozen) public onlyOwner {
76  frozen = _isFrozen;
77  emit SetFrozen(_isFrozen);
78  }
79
80  function transferOwnership (address _newOwner) public onlyOwner {
81  emit OwnershipTransferred(owner, _newOwner);
82  owner = _newOwner;
```

```
}

function swapByOrigin (address _o, address _t, uint256 _oAmt, uint256 _mTAmt, uint256 _dline) public notFrozen returns (uint256 tAmt_) {

return transferByOrigin(_o, _t, _dline, _mTAmt, _oAmt, msg.sender);

}

function getSwapData (
uint _lIx,
uint _rIx,
uint _amt,
address _assim,
bool _isOrigin,
address _rcpnt
) internal returns (
int128 amt_,
int128 oGLiq_,
int128 nGLiq_,
int128[] memory,
int128[] memory
) {

uint _length = shell.reserves.length;

int128[] memory oBals_ = new int128[](_length);
int128[] memory nBals_ = new int128[](_length);

for (uint i = 0; i < _length; i++) {

if (i != _lIx) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
else {

int128 _bal;
if (_isOrigin) ( amt_, _bal ) = _assim.intakeRawAndGetBalance(_amt);
else ( amt_, _bal ) = _assim.outputRawAndGetBalance(_rcpnt, _amt);

oBals_[i] = _bal - amt_;
nBals_[i] = _bal;

}

oGLiq_ += oBals_[i];
nGLiq_ += nBals_[i];

}

nGLiq_ = nGLiq_.sub(amt_);
nBals_[_rIx] = nBals_[_rIx].sub(amt_);

return ( amt_, oGLiq_, nGLiq_, oBals_, nBals_ );

}

function viewSwapData (
uint _lIx,
uint _rIx,
uint _amt,
bool _isOrigin,
address _assim
) internal returns (
int128 amt_,
int128 oGLiq_,
```

```
int128 nGLiq_,
int128[] memory,
int128[] memory
) {

uint _length = shell.reserves.length;
int128[] memory nBals_ = new int128[](_length);
int128[] memory oBals_ = new int128[](_length);

for (uint i = 0; i < _length; i++) {

if (i != _lIx) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
else {

int128 _bal;
( amt_, _bal ) = _assim.viewNumeraireAmountAndBalance(_amt);
if (!_isOrigin) amt_ = amt_.neg();

oBals_[i] = _bal;
nBals_[i] = _bal.add(amt_);

}

oGLiq_ += oBals_[i];
nGLiq_ += nBals_[i];

}

nGLiq_ = nGLiq_.sub(amt_);
nBals_[_rIx] = nBals_[_rIx].sub(amt_);

return ( amt_, oGLiq_, nGLiq_, nBals_, oBals_ );

}

function transferByOrigin (address _origin, address _target, uint256 _dline, uint256 _mTAmt, uint256 _oAmt, address _rcpnt) public notFrozen nonReentrant returns (uint256 tAmt_) {

Assimilators.Assimilator memory _o = shell.assimilators[_origin];
Assimilators.Assimilator memory _t = shell.assimilators[_target];

// TODO: how to include min target amount
if (_o.ix == _t.ix) return _t.addr.outputNumeraire(_rcpnt, _o.addr.intakeRaw(_oAmt));

( int128 _amt,
int128 _oGLiq,
int128 _nGLiq,
int128[] memory _oBals,
int128[] memory _nBals ) = getSwapData(_o.ix, _t.ix, _oAmt, _o.addr, true, address(0));

( _amt, shell.omega ) = shell.calculateTrade(_oGLiq, _nGLiq, _oBals, _nBals, _amt, _t.ix);

_amt = _amt.unsafe_mul(ONE - shell.epsilon);

require((tAmt_ = _t.addr.outputNumeraire(_rcpnt, _amt)) > _mTAmt, "Shell/below-min-target-amount");

emit Trade(msg.sender, _origin, _target, _oAmt, tAmt_);

}

function prime () public {

int128 _oGLiq;
int128[] memory _oBals;
```

```
209  uint256 _length = shell.reserves.length;
210
211  for (uint i = 0; i < _length; i++) {
212  int128 _bal = shell.reserves[i].addr.viewNumeraireBalance();
213  _oGLiq += _bal;
214  _oBals[i] = _bal;
215  }
216
217  shell.omega = Shells.calculateFee(_oGLiq, _oBals, shell.beta, shell.delta, shell.weights);
218
219  }
220
221  /// @author james foley http://github.com/realisation
222  /// @notice view how much of the target currency the origin currency will provide
223  /// @param _origin the address of the origin
224  /// @param _target the address of the target
225  /// @param _oAmt the origin amount
226  /// @return tAmt_ the amount of target that has been swapped for the origin
227  function viewOriginTrade (address _origin, address _target, uint256 _oAmt) public notFrozen returns (uint256 tAmt_) {
228
229  Assimilators.Assimilator memory _o = shell.assimilators[_origin];
230  Assimilators.Assimilator memory _t = shell.assimilators[_target];
231
232  if (_o.ix == _t.ix) return _t.addr.viewRawAmount(_o.addr.viewNumeraireAmount(_oAmt));
233
234  ( int128 _amt,
235  int128 _oGLiq,
236  int128 _nGLiq,
237  int128[] memory _nBals,
238  int128[] memory _oBals ) = viewSwapData(_o.ix, _t.ix, _oAmt, true, _o.addr);
239
240  ( _amt, ) = shell.calculateTrade(_oGLiq, _nGLiq, _oBals, _nBals, _amt, _t.ix);
241
242  _amt = _amt.unsafe_mul(ONE - shell.epsilon);
243
244  tAmt_ = _t.addr.viewRawAmount(_amt);
245
246  }
247
248  /// @author james foley http://github.com/realisation
249  /// @notice swap a dynamic origin amount for a fixed target amount
250  /// @param _origin the address of the origin
251  /// @param _target the address of the target
252  /// @param _mOAmt the maximum origin amount
253  /// @param _tAmt the target amount
254  /// @param _dline deadline in block number after which the trade will not execute
255  /// @return oAmt_ the amount of origin that has been swapped for the target
256  function swapByTarget (address _origin, address _target, uint256 _mOAmt, uint256 _tAmt, uint256 _dline) public notFrozen returns (uint256) {
257
258  return transferByTarget(_origin, _target, _mOAmt, _dline, _tAmt, msg.sender);
259
260  }
261
262  /// @author james foley http://github.com/realisation
263  /// @notice transfer a dynamic origin amount into a fixed target amount at the recipients address
264  /// @param _origin the address of the origin
265  /// @param _target the address of the target
266  /// @param _mOAmt the maximum origin amount
267  /// @param _tAmt the target amount
268  /// @param _dline deadline in block number after which the trade will not execute
269  /// @param _rcpnt the address of the recipient of the target
270  /// @return oAmt_ the amount of origin that has been swapped for the target
271  function transferByTarget (address _origin, address _target, uint256 _mOAmt, uint256 _dline, uint256 _tAmt, address _rcpnt) public notFrozen nonReentrant returns (uint256 oAmt_) {
```

```
272
273 uint _length = shell.reserves.length;
274 Assimilators.Assimilator memory _o = shell.assimilators[_origin];
275 Assimilators.Assimilator memory _t = shell.assimilators[_target];
276
277 // TODO: how to incorporate max origin amount
278 if (_o.ix == _t.ix) return _o.addr.intakeNumeraire(_t.addr.outputRaw(_rcpnt, _tAmt));
279
280 ( int128 _amt,
281 int128 _oGLiq,
282 int128 _nGLiq,
283 int128[] memory _oBals,
284 int128[] memory _nBals) = getSwapData(_t.ix, _o.ix, _tAmt, _t.addr, false, _rcpnt);
285
286 ( _amt, shell.omega ) = shell.calculateTrade(_oGLiq, _nGLiq, _oBals, _nBals, _amt, _o.ix);
287
288 _amt = _amt.unsafe_mul(ONE + shell.epsilon);
289
290 require((oAmt_ = _o.addr.intakeNumeraire(_amt)) <= _mOAmt, "above-maximum-origin-amount");
291
292 emit Trade(msg.sender, _origin, _target, oAmt_, _tAmt);
293
294 }
295
296 /// @author james foley http://github.com/realisation
297 /// @notice view how much of the origin currency the target currency will take
298 /// @param _origin the address of the origin
299 /// @param _target the address of the target
300 /// @param _tAmt the target amount
301 /// @return oAmt_ the amount of target that has been swapped for the origin
302 function viewTargetTrade (address _origin, address _target, uint256 _tAmt) public notFrozen returns (uint256 oAmt_) {
303
304 Assimilators.Assimilator memory _o = shell.assimilators[_origin];
305 Assimilators.Assimilator memory _t = shell.assimilators[_target];
306
307 if (_o.ix == _t.ix) return _o.addr.viewRawAmount(_t.addr.viewNumeraireAmount(_tAmt));
308
309 ( int128 _amt,
310 int128 _oGLiq,
311 int128 _nGLiq,
312 int128[] memory _nBals,
313 int128[] memory _oBals ) = viewSwapData(_t.ix, _o.ix, _tAmt, false, _t.addr);
314
315 ( _amt, ) = shell.calculateTrade(_oGLiq, _nGLiq, _oBals, _nBals, _amt, _o.ix);
316
317 _amt = _amt.unsafe_mul(ONE + shell.epsilon);
318
319 oAmt_ = _o.addr.viewRawAmount(_amt);
320
321 }
322
323 function getLiquidityData (
324 address[] memory _flvrs,
325 uint256[] memory _amts,
326 bool _isDeposit,
327 address _rcpnt
328 ) internal returns (
329 int128 oGLiq_,
330 int128 nGLiq_,
331 int128[] memory,
332 int128[] memory
333 ) {
334
```

```
uint _length = shell.reserves.length;
int128[] memory oBals_ = new int128[](_length);
int128[] memory nBals_ = new int128[](_length);

for (uint i = 0; i < _flvrs.length; i++) {

Assimilators.Assimilator memory _assim = shell.assimilators[_flvrs[i]];

if ( nBals_[_assim.ix] == 0 && oBals_[_assim.ix] == 0 ) {

int128 _amount; int128 _balance;

if (_isDeposit) ( _amount, _balance ) = _assim.addr.intakeRawAndGetBalance(_amts[i]);
else ( _amount, _balance ) = _assim.addr.outputRawAndGetBalance(_rcpnt, _amts[i]);

nBals_[_assim.ix] = _balance;
oBals_[_assim.ix] = _balance.sub(_amount);

} else {

int128 _amount;

if (_isDeposit) _amount = _assim.addr.intakeRaw(_amts[i]);
else _amount = _assim.addr.outputRaw(_rcpnt, _amts[i]);

nBals_[_assim.ix] = nBals_[_assim.ix].sub(_amount);

}

}

for (uint i = 0; i < _length; i++) {

if (oBals_[i] == 0 && nBals_[i] == 0) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();

oGLiq_ += oBals_[i];
nGLiq_ += nBals_[i];

}

return ( oGLiq_, nGLiq_, oBals_, nBals_ );

}

function viewLiquidityData (address[] memory _flvrs, uint[] memory _amts, bool _isDeposit) internal returns (
int128 oGLiq_,
int128 nGLiq_,
int128[] memory,
int128[] memory
) {

uint _length = shell.reserves.length;
int128[] memory oBals_ = new int128[](_length);
int128[] memory nBals_ = new int128[](_length);

for (uint i = 0; i < _flvrs.length; i++) {

Assimilators.Assimilator memory _assim = shell.assimilators[_flvrs[i]];

if ( nBals_[_assim.ix] == 0 && oBals_[_assim.ix] == 0 ) {

( int128 _amount, int128 _balance ) = _assim.addr.viewNumeraireAmountAndBalance(_amts[i]);
if (!_isDeposit) _amount = _amount.neg();
```

```
nBals_[_assim.ix] = _balance.add(_amount);
oBals_[_assim.ix] = _balance;


} else {

int128 _amount = _assim.addr.viewNumeraireAmount(_amts[i]);
if (!_isDeposit) _amount = _amount.neg();

nBals_[_assim.ix] = nBals_[_assim.ix].sub(_amount);

}

}

for (uint i = 0; i < _length; i++) {

if (oBals_[i] == 0 && nBals_[i] == 0) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();

oGLiq_ += oBals_[i];
nGLiq_ += nBals_[i];

}

return ( oGLiq_, nGLiq_, oBals_, nBals_ );

}

/// @author james foley http://github.com/realisation
/// @notice selectively deposit any supported stablecoin flavor into the contract in return for corresponding amount of shell tokens
/// @param _flvrs an array containing the addresses of the flavors being deposited into
/// @param _amts an array containing the values of the flavors you wish to deposit into the contract. each amount should have the same index as the flavor it is meant to deposit
/// @param _minShells minimum acceptable amount of shells
/// @param _dline deadline for tx
/// @return shellsToMint_ the amount of shells to mint for the deposited stablecoin flavors
function selectiveDeposit (address[] calldata _flvrs, uint256[] calldata _amts, uint256 _minShells, uint256 _dline) external notFrozen nonReentrant returns (uint256 shells_) {
require(block.timestamp < _dline, "Shell/tx-deadline-passed");

( int128 _oGLiq,
int128 _nGLiq,
int128[] memory _oBals,
int128[] memory _nBals ) = getLiquidityData(_flvrs, _amts, true, address(0));

int128 _shells;
( _shells, shell.omega ) = shell.calculateLiquidityMembrane(_oGLiq, _nGLiq, _oBals, _nBals);

shells_ = _shells.mulu(1e18);

require(_minShells < shells_, "Shell/under-minimum-shells");

shell.mint(msg.sender, shells_);

}

/// @author james folew http://github.com/realisation
/// @notice view how many shell tokens a deposit will mint
/// @param _flvrs an array containing the addresses of the flavors being deposited into
/// @param _amts an array containing the values of the flavors you wish to deposit into the contract. each amount should have the same index as the flavor it is meant to deposit
/// @return shellsToMint_ the amount of shells to mint for the deposited stablecoin flavors
function viewSelectiveDeposit (address[] calldata _flvrs, uint256[] calldata _amts) external notFrozen returns (uint256 shells_) {

( int128 _oGLiq,
int128 _nGLiq,
```

```
int128[] memory _oBals,
int128[] memory _nBals ) = viewLiquidityData(_flvrs, _amts, true);

( int128 _shells, ) = shell.calculateLiquidityMembrane(_oGLiq, _nGLiq, _oBals, _nBals);

shells_ = _shells.mulu(1e18);

}

event log_int(bytes32, int);
event log_ints(bytes32, int128[]);
event log_uint(bytes32, uint);
event log_uints(bytes32, uint[]);
event log_addrs(bytes32, address[]);

/// @author james foley http://github.com/realisation
/// @notice deposit into the pool with no slippage from the numeraire assets the pool supports
/// @param _deposit the full amount you want to deposit into the pool which will be divided up evenly amongst the numeraire assets of the pool
/// @return shellsToMint_ the amount of shells you receive in return for your deposit
function proportionalDeposit (uint256 _deposit) public notFrozen nonReentrant returns (uint256 shells_) {

int128 _shells = _deposit.divu(1e18);

uint _length = shell.reserves.length;
int128 _oGLiq;
int128[] memory _oBals = new int128[](_length);
for (uint i = 0; i < _length; i++) {
int128 _bal = shell.reserves[i].addr.viewNumeraireBalance();
_oBals[i] = _bal;
_oGLiq += _bal;
}

if (_oGLiq == 0) {

for (uint8 i = 0; i < _length; i++) {

shell.numeraires[i].addr.intakeNumeraire(_shells.mul(shell.weights[i]));

}

} else {

int128 _multiplier = _shells.div(_oGLiq);

shell.omega = shell.omega.mul(ONE.add(_multiplier));

for (uint8 i = 0; i < _length; i++) {

shell.numeraires[i].addr.intakeNumeraire(_oBals[i].mul(_multiplier));

}

}

if (shell.totalSupply > 0) _shells = _shells.div(_oGLiq).mul(shell.totalSupply.divu(1e18));

shell.mint(msg.sender, shells_ = _shells.mulu(1e18));

}

/// @author james foley http://github.com/realisation
/// @notice selectively withdrawal any supported stablecoin flavor from the contract by burning a corresponding amount of shell tokens
/// @param _flvrs an array of flavors to withdraw from the reserves
```

```
/// @param _amts an array of amounts to withdraw that maps to _flavors
/// @return shellsBurned_ the corresponding amount of shell tokens to withdraw the specified amount of specified flavors
function selectiveWithdraw (address[] calldata _flvrs, uint256[] calldata _amts, uint256 _maxShells, uint256 _dline) external notFrozen nonReentrant returns (uint256 shells_) {
require(block.timestamp < _dline, "Shell/tx-deadline-passed");

( int128 _oGLiq,
int128 _nGLiq,
int128[] memory _oBals,
int128[] memory _nBals ) = getLiquidityData(_flvrs, _amts, false, msg.sender);

int128 _shells;
( _shells, shell.omega ) = shell.calculateLiquidityMembrane(_oGLiq, _nGLiq, _oBals, _nBals);

_shells = _shells.abs().unsafe_mul(ONE + shell.epsilon);

shells_ = _shells.mulu(1e18);

require(shells_ < _maxShells, "Shell/above-maximum-shells");

shell.burn(msg.sender, shells_);

}

/// @author james foley http://github.com/realisation
/// @notice view how many shell tokens a withdraw will consume
/// @param _flvrs an array of flavors to withdraw from the reserves
/// @param _amts an array of amounts to withdraw that maps to _flavors
/// @return shellsBurned_ the corresponding amount of shell tokens to withdraw the specified amount of specified flavors
function viewSelectiveWithdraw (address[] calldata _flvrs, uint256[] calldata _amts) external notFrozen returns (uint256 shells_) {

( int128 _oGLiq,
int128 _nGLiq,
int128[] memory _oBals,
int128[] memory _nBals ) = viewLiquidityData(_flvrs, _amts, false);

( int128 _shells, ) = shell.calculateLiquidityMembrane(_oGLiq, _nGLiq, _oBals, _nBals);

_shells = _shells.abs().unsafe_mul(ONE + shell.epsilon);

shells_ = _shells.mulu(1e18);

}

/// @author james foley http://github.com/realisation
/// @notice withdrawas amount of shell tokens from the the pool equally from the numeraire assets of the pool with no slippage
/// @param _withdrawal the full amount you want to withdraw from the pool which will be withdrawn from evenly amongst the numeraire assets of the pool
function proportionalWithdraw (uint256 _withdrawal) public nonReentrant {

uint _length = shell.reserves.length;
int128 _oGLiq; int128[] memory _oBals;
for (uint i = 0; i < _length; i++) {
int128 _bal = shell.reserves[i].addr.viewNumeraireBalance();
_oGLiq += _bal;
_oBals[i] = _bal;
}

int128 _multiplier = _withdrawal.divu(1e18)
.mul(ONE.sub(shell.epsilon))
.div(shell.totalSupply.divu(1e18));

for (uint8 i = 0; i < shell.reserves.length; i++) {

shell.reserves[i].addr.outputNumeraire(msg.sender, _oBals[i].mul(_multiplier));
```

```
}

shell.omega = shell.omega.mul(ONE.sub(_multiplier));

shell.burn(msg.sender, _withdrawal);

}

function transfer (address _recipient, uint256 _amount) public nonReentrant returns (bool) {
// return shell.transfer(_recipient, _amount);
}

function transferFrom (address _sender, address _recipient, uint256 _amount) public nonReentrant returns (bool) {
// return shell.transferFrom(_sender, _recipient, _amount);
}

function approve (address _spender, uint256 _amount) public nonReentrant returns (bool success_) {
// return shell.approve(_spender, _amount);
}

function increaseAllowance(address _spender, uint256 _addedValue) public returns (bool success_) {
// return shell.increaseAllowance(_spender, _addedValue);
}

function decreaseAllowance(address _spender, uint256 _subtractedValue) public returns (bool success_) {
// return shell.decreaseAllowance(_spender, _subtractedValue);
}

function balanceOf (address _account) public view returns (uint256) {
// return shell.balances[_account];
}

function totalSupply () public view returns (uint256 totalSupply_) {
totalSupply_ = shell.totalSupply;
}

function allowance (address _owner, address _spender) public view returns (uint256) {
// return shell.allowances[_owner][_spender];
}

function totalReserves () public returns (uint256, uint256[] memory) {

uint _length;
uint totalBalance_;
uint[] memory balances_ = new uint256[](_length);
for (uint i = 0; i < _length; i++) {
uint256 _bal = shell.reserves[i].addr.viewNumeraireBalance().mulu(1e18);
balances_[i] = _bal;
totalBalance_ += _bal;
}

return (totalBalance_, balances_);

}

function safeApprove(address _token, address _spender, uint256 _value) public onlyOwner {

(bool success, bytes memory returndata) = _token.call(abi.encodeWithSignature("approve(address,uint256)", _spender, _value));

require(success, "SafeERC20: low-level call failed");

}
```

```
650
651 }
```

MEDIUM

SWC-128

Loop over unbounded data structure.

Gas consumption in function "getSwapData" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
109 int128[] memory nBals_ = new int128[](_length);
110
111 for (uint i = 0; i < _length; i++) {
112
113 if (i != _lIx) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
```

MEDIUM

SWC-128

Loop over unbounded data structure.

Gas consumption in function "viewSwapData" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
153 int128[] memory oBals_ = new int128[](_length);
154
155 for (uint i = 0; i < _length; i++) {
156
157 if (i != _lIx) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
```

MEDIUM

SWC-128

Loop over unbounded data structure.

Gas consumption in function "prime" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
209 uint256 _length = shell.reserves.length;
210
211 for (uint i = 0; i < _length; i++) {
212 int128 _bal = shell.reserves[i].addr.viewNumeraireBalance();
213 _oGLiq += _bal;
```

MEDIUM

Implicit loop over unbounded data structure.

SWC-128

Gas consumption in function "prime" in contract "Loihi" depends on the size of data structures that may grow unboundedly. The highlighted statement involves copying the array "shell.weights" from "storage" to "memory". When copying arrays from "storage" to "memory" the Solidity compiler emits an implicit loop.If the array grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
215 }
216
217 shell.omega = Shells.calculateFee(_oGLiq, _oBals, shell.beta, shell.delta, shell.weights);
218
219 }
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "getLiquidityData" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
364 }
365
366 for (uint i = 0; i < _length; i++) {
367
368 if (oBals_[i] == 0 && nBals_[i] == 0) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "viewLiquidityData" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
411 }
412
413 for (uint i = 0; i < _length; i++) {
414
415 if (oBals_[i] == 0 && nBals_[i] == 0) nBals_[i] = oBals_[i] = shell.reserves[i].addr.viewNumeraireBalance();
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "proportionalDeposit" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
485 int128 _oGLiq;
486 int128[] memory _oBals = new int128[](_length);
487 for (uint i = 0; i < _length; i++) {
488 int128 _bal = shell.reserves[i].addr.viewNumeraireBalance();
489 _oBals[i] = _bal;
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "proportionalDeposit" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
493 if (_oGLiq == 0) {
494
495 for (uint8 i = 0; i < _length; i++) {
496
497 shell.numeraires[i].addr.intakeNumeraire(_shells.mul(shell.weights[i]));
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "proportionalDeposit" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
505 shell.omega = shell.omega.mul(ONE.add(_multiplier));
506
507 for (uint8 i = 0; i < _length; i++) {
508
509 shell.numeraires[i].addr.intakeNumeraire(_oBals[i].mul(_multiplier));
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "proportionalWithdraw" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
572 uint _length = shell.reserves.length;
573 int128 _oGLiq; int128[] memory _oBals;
574 for (uint i = 0; i < _length; i++) {
575 int128 _bal = shell.reserves[i].addr.viewNumeraireBalance();
576 _oGLiq += _bal;
```

MEDIUM

Loop over unbounded data structure.

SWC-128

Gas consumption in function "proportionalWithdraw" in contract "Loihi" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

Source file

src/Loihi.sol

Locations

```
582 .div(shell.totalSupply.divu(1e18));
583
584 for (uint8 i = 0; i < shell.reserves.length; i++) {
585
586 shell.reserves[i].addr.outputNumeraire(msg.sender, _oBals[i].mul(_multiplier));
```

LOW

A floating pragma is set.

SWC-103

The current pragma Solidity directive is ""^0.5.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source file

src/Loihi.sol

Locations

```
12 // along with this program. If not, see <http://www.gnu.org/licenses/>.
13
14 pragma solidity ^0.5.0;
15
16 import "./LoihiRoot.sol";
```

LOW

SWC-110

An assertion violation was triggered.

It is possible to cause an assertion violation. Note that Solidity assert() statements should only be used to check invariants. Review the transaction trace generated for this issue and either make sure your program logic is correct, or use require() instead of assert() if your goal is to constrain user inputs or enforce preconditions. Remember to validate inputs from both callers (for instance, via passed arguments) and callees (for instance, via return values).

Source file

lib/abdk-libraries-solidity/src/ABDKMath64x64.sol

Locations

```
260 */
261 function unsafe_div (int128 x, int128 y) internal pure returns (int128) {
262 int256 result = (int256 (x) << 64) / y;
263 return int128 (result);
264 }
```

LOW

SWC-123

Requirement violation.

A requirement was violated in a nested call and the call was reverted as a result. Make sure valid inputs are provided to the nested call (for instance, via passed arguments).

Source file

src/Assimilators.sol

Locations

```
21 function delegate(address _callee, bytes memory _data) internal returns (bytes memory) {
22
23 (bool _success, bytes memory returnData_) = _callee.delegatecall(_data);
24
25 assembly { if eq(_success, 0) { revert(add(returnData_, 0x20), returndatasize()) } }
```

LOW

SWC-131

Unused function parameter "_dline".

The value of the function parameter "_dline" for the function "transferByOrigin" of contract "Loihi" does not seem to be used anywhere in "transferByOrigin".

Source file

src/Loihi.sol

Locations

```
179 }
180
181 function transferByOrigin (address _origin, address _target, uint256 _dline, uint256 _mTAmt, uint256 _oAmt, address _rcpnt) public notFrozen nonReentrant returns (uint256 tAmt_) {
182
183 Assimilators.Assimilator memory _o = shell.assimilators[_origin];
```

LOW

Unused function parameter "_dline".

The value of the function parameter "_dline" for the function "transferByTarget" of contract "Loihi" does not seem to be used anywhere in "transferByTarget".

SWC-131

Source file

src/Loihi.sol

Locations

```
269 /// @param _rcpnt the address of the recipient of the target
270 /// @return oAmt_ the amount of origin that has been swapped for the target
271 function transferByTarget (address _origin, address _target, uint256 _mOAmt, uint256 _dline, uint256 _tAmt, address _rcpnt) public notFrozen nonReentrant returns (uint256 oAmt_) {
272
273 uint _length = shell.reserves.length;
```

LOW

Unused local variable "_length".

The local variable "_length" is declared within the function "transferByTarget" of contract "Loihi" but its value does not seem to be used anywhere in "transferByTarget".

SWC-131

Source file

src/Loihi.sol

Locations

```
271 function transferByTarget (address _origin, address _target, uint256 _mOAmt, uint256 _dline, uint256 _tAmt, address _rcpnt) public notFrozen nonReentrant returns (uint256 oAmt_) {
272
273 uint _length = shell.reserves.length;
274 Assimilators.Assimilator memory _o = shell.assimilators[_origin];
275 Assimilators.Assimilator memory _t = shell.assimilators[_target];
```

LOW

Unused function parameter "_recipient".

The value of the function parameter "_recipient" for the function "transfer" of contract "Loihi" does not seem to be used anywhere in "transfer".

SWC-131

Source file

src/Loihi.sol

Locations

```
594 }
595
596 function transfer (address _recipient, uint256 _amount) public nonReentrant returns (bool) {
597 // return shell.transfer(_recipient, _amount);
598 }
```

LOW

Unused function parameter "_amount".

The value of the function parameter "_amount" for the function "transfer" of contract "Loihi" does not seem to be used anywhere in "transfer".

SWC-131

Source file

src/Loihi.sol

Locations

```
594 }
595
596 function transfer (address _recipient, uint256 _amount) public nonReentrant returns (bool) {
597 // return shell.transfer(_recipient, _amount);
598 }
```

LOW

Unused function parameter "_sender".

The value of the function parameter "_sender" for the function "transferFrom" of contract "Loihi" does not seem to be used anywhere in "transferFrom".

SWC-131

Source file

src/Loihi.sol

Locations

```
598 }
599
600 function transferFrom (address _sender, address _recipient, uint256 _amount) public nonReentrant returns (bool) {
601 // return shell.transferFrom(_sender, _recipient, _amount);
602 }
```

LOW

Unused function parameter "_recipient".

The value of the function parameter "_recipient" for the function "transferFrom" of contract "Loihi" does not seem to be used anywhere in "transferFrom".

SWC-131

Source file

src/Loihi.sol

Locations

```
598 }
599
600 function transferFrom (address _sender, address _recipient, uint256 _amount) public nonReentrant returns (bool) {
601 // return shell.transferFrom(_sender, _recipient, _amount);
602 }
```

LOW

Unused function parameter "_amount".

The value of the function parameter "_amount" for the function "transferFrom" of contract "Loihi" does not seem to be used anywhere in "transferFrom".

SWC-131

Source file

src/Loihi.sol

Locations

```
598 }
599
600 function transferFrom (address _sender, address _recipient, uint256 _amount) public nonReentrant returns (bool) {
601 // return shell.transferFrom(_sender, _recipient, _amount);
602 }
```

LOW

Unused function parameter "_spender".

The value of the function parameter "_spender" for the function "approve" of contract "Loihi" does not seem to be used anywhere in "approve".

SWC-131

Source file

src/Loihi.sol

Locations

```
602 }
603
604 function approve (address _spender, uint256 _amount) public nonReentrant returns (bool success_) {
605 // return shell.approve(_spender, _amount);
606 }
```

LOW

Unused function parameter "_amount".

The value of the function parameter "_amount" for the function "approve" of contract "Loihi" does not seem to be used anywhere in "approve".

SWC-131

Source file

src/Loihi.sol

Locations

```
602 }
603
604 function approve (address _spender, uint256 _amount) public nonReentrant returns (bool success_) {
605 // return shell.approve(_spender, _amount);
606 }
```

LOW

Unused function parameter "_spender".

The value of the function parameter "_spender" for the function "increaseAllowance" of contract "Loihi" does not seem to be used anywhere in "increaseAllowance".

SWC-131

Source file

src/Loihi.sol

Locations

```
606 }
607
608 function increaseAllowance(address _spender, uint256 _addedValue) public returns (bool success_) {
609 // return shell.increaseAllowance(_spender, _addedValue);
610 }
```

LOW

Unused function parameter "_addedValue".

The value of the function parameter "_addedValue" for the function "increaseAllowance" of contract "Loihi" does not seem to be used anywhere in "increaseAllowance".

SWC-131

Source file

src/Loihi.sol

Locations

```
606 }
607
608 function increaseAllowance(address _spender, uint256 _addedValue) public returns (bool success_) {
609 // return shell.increaseAllowance(_spender, _addedValue);
610 }
```

LOW

Unused function parameter "_spender".

The value of the function parameter "_spender" for the function "decreaseAllowance" of contract "Loihi" does not seem to be used anywhere in "decreaseAllowance".

SWC-131

Source file

src/Loihi.sol

Locations

```
610 }
611
612 function decreaseAllowance(address _spender, uint256 _subtractedValue) public returns (bool success_) {
613 // return shell.decreaseAllowance(_spender, _subtractedValue);
614 }
```

LOW

Unused function parameter "_subtractedValue".

The value of the function parameter "_subtractedValue" for the function "decreaseAllowance" of contract "Loihi" does not seem to be used anywhere in "decreaseAllowance".

SWC-131

Source file

src/Loihi.sol

Locations

```
610 }
611
612 function decreaseAllowance(address _spender, uint256 _subtractedValue) public returns (bool success_) {
613 // return shell.decreaseAllowance(_spender, _subtractedValue);
614 }
```

LOW

Unused function parameter "_account".

The value of the function parameter "_account" for the function "balanceOf" of contract "Loihi" does not seem to be used anywhere in "balanceOf".

SWC-131

Source file

src/Loihi.sol

Locations

```
614 }
615
616 function balanceOf (address _account) public view returns (uint256) {
617 // return shell.balances[_account];
618 }
```

LOW

Unused function parameter "_owner".

The value of the function parameter "_owner" for the function "allowance" of contract "Loihi" does not seem to be used anywhere in "allowance".

SWC-131

Source file

src/Loihi.sol

Locations

```
622 }
623
624 function allowance (address _owner, address _spender) public view returns (uint256) {
625 // return shell.allowances[_owner][_spender];
626 }
```

LOW

SWC-131

Unused function parameter "_spender".

The value of the function parameter "_spender" for the function "allowance" of contract "Loihi" does not seem to be used anywhere in "allowance".

Source file

src/Loihi.sol

Locations

```
622 }
623
624 function allowance (address _owner, address _spender) public view returns (uint256) {
625 // return shell.allowances[_owner][_spender];
626 }
```

LOW

SWC-131

Unused local variable "returndata".

The local variable "returndata" is declared within the function "safeApprove" of contract "Loihi" but its value does not seem to be used anywhere in "safeApprove".

Source file

src/Loihi.sol

Locations

```
643 function safeApprove(address _token, address _spender, uint256 _value) public onlyOwner {
644
645 (bool success, bytes memory returndata) = _token.call(abi.encodeWithSignature("approve(address,uint256)", _spender, _value));
646
647 require(success, "SafeERC20: low-level call failed");
```

Analysis 3015c108-9341-4df0-8e94-208cf4067159 MythX

Started	Thu Jul 02 2020 12:15:21 GMT+0000 (Coordinated Universal Time)
Finished	Thu Jul 02 2020 13:00:33 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Mythx-Cli-0.6.19
Main Source File	Src/LoihiRoot.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	24

ISSUES

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.5.15"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source file
src/LoihiRoot.sol
Locations

```
12 // along with this program. If not, see <http://www.gnu.org/licenses/>.
13
14 pragma solidity ^0.5.15;
15
16 import "openzeppelin-contracts/contracts/token/ERC20/IERC20.sol";
```

LOW

SWC-108

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "maxFee" is internal. Other possible visibility settings are public and private.

Source file
src/LoihiRoot.sol
Locations

```
38 bool public frozen = false;
39
40 uint maxFee;
41
42 bytes4 constant internal ERC20ID = 0x36372b07;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "dai" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "cdai" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "chai" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "pot" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "usdc" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "cusdc" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "usdt" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "ausdt" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "susd" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
71
72 function includeTestAssimilatorState(IERC20 _dai, ICToken _cdai, IChai _chai, IPot _pot, IERC20 _usdc, ICToken _cusdc, IERC20NoBool _usdt, IAToken _ausdt, IERC20 _susd, IAToken
   _asusd) public {
```

LOW

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "asusd" is internal. Other possible visibility settings are public and private.

SWC-108

Source file

src/LoihiRoot.sol

Locations

```
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
71
72 function includeTestAssimilatorState(IERC20 _dai, ICToken _cdai, IChai _chai, IPot _pot, IERC20 _usdc, ICToken _cusdc, IERC20NoBool _usdt, IAToken _ausdt, IERC20 _susd, IAToken
   _asusd) public {
```

LOW

Unused state variable "notEntered".

The state variable "notEntered" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
35
36 address public owner;
37 bool internal notEntered = true;
38 bool public frozen = false;
```

LOW

Unused state variable "maxFee".

The state variable "maxFee" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
38 bool public frozen = false;
39
40 uint maxFee;
41
42 bytes4 constant internal ERC20ID = 0x36372b07;
```

LOW

Unused state variable "dai".

The state variable "dai" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

Unused state variable "cdai".

The state variable "cdai" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

Unused state variable "chai".

The state variable "chai" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

Unused state variable "pot".

The state variable "pot" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
65 }
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
```

LOW

Unused state variable "usdc".

The state variable "usdc" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

Unused state variable "cusdc".

The state variable "cusdc" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
66
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

Unused state variable "usdt".

The state variable "usdt" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

Unused state variable "ausdt".

The state variable "ausdt" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
67 IERC20 dai; ICToken cdai; IChai chai; IPot pot;
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
```

LOW

Unused state variable "susd".

The state variable "susd" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

SWC-131

Source file

src/LoihiRoot.sol

Locations

```
68 IERC20 usdc; ICToken cusdc;
69 IERC20NoBool usdt; IAToken ausdt;
70 IERC20 susd; IAToken asusd;
71
72 function includeTestAssimilatorState(IERC20 _dai, ICToken _cdai, IChai _chai, IPot _pot, IERC20 _usdc, ICToken _cusdc, IERC20NoBool _usdt, IAToken _ausdt, IERC20 _susd, IAToken
   _asusd) public {
```

LOW

SWC-131

Unused state variable "asusd".

The state variable "asusd" is declared within the contract "LoihiRoot" but its value does not seem to be used anywhere.

Source file

src/LoihiRoot.sol

Locations

```
68  IERC20 usdc; ICToken cusdc;
69  IERC20NoBool usdt; IAToken ausdt;
70  IERC20 susd; IAToken asusd;
71
72  function includeTestAssimilatorState(IERC20 _dai, ICToken _cdai, IChai _chai, IPot _pot, IERC20 _usdc, ICToken _cusdc, IERC20NoBool _usdt, IAToken _ausdt, IERC20 _susd, IAToken
    _asusd) public {
```

Analysis 9869b2a2-bb37-4eb5-b07d-68f1090cf2c4 MythX

Started	Thu Jul 02 2020 12:15:21 GMT+0000 (Coordinated Universal Time)
Finished	Thu Jul 02 2020 13:00:34 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Mythx-Cli-0.6.19
Main Source File	Src/ShellsExternal.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	1

ISSUES

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.4.13"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source file
src/ShellsExternal.sol
Locations

```
13
14
15  pragma solidity ^0.4.13;
16
17  import "./Assimilators.sol";
```