0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

EMP Money

January 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

The report has been prepared for the EMP Money team. The project website is https://emp.money/. The audited project is a fork of the Tomb Finance Project. The purpose of the audit was to ensure that no issues were introduced with the changes to the original code and that known vulnerabilities (e.g. circumventing the protocol's fee system) are fixed.

Name	EMP Money
Audit date	2022-01-26 - 2022-01-26
Language	Solidity
Platform	Binance Smart Chain

Contracts checked

Name	Address
EBond	0x7099A19Da2f17BC85193B1f0e9091dF014A5D520
EmpRewardPool	0xa7097828dc57E50A5c83005906C3cF8c453dfA79
Emp	0x3b248CEfA87F836a4e6f6d6c9b42991b88Dc1d58
Zapper	0x1732Bb86dcd3D29e041Aa88fF8fee947c8ABAEd2
EShare	0xDB20F6A8665432CE895D724b417f77EcAC956550
TaxOffice	0xc34aC3fc955085AC23238F55a4c6a34F554C3B47
Oracle	0x0Fe57361B0E3Fc7F61972BD839Ddaa8Da3E691D2
EShareRewardPool	0x97a68a7949EE30849D273b0c4450314ae26235b1
Boardroom	0x7a51c848babaeedc58dc89556583b06f6f7dbcf3
TaxOfficeV2	0x12A9691B3BD61f0d235cf95676D6a7a555164768
Treasury	0xd3DD99430a7C6818F8C848eCffeD527d38505bb0
EmpGenesisRewardPool	0x1F5659CDa58B245cE19ddD499c81eB0c8A29da1f
TaxOracle	0x973bBDfFD30429d820465b2c2Dc9CA79F1f48Eb8

Procedure

We perform our audit according to the following procedure:

Automated analysis

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
- Manual verification (reject or confirm) all the issues found by the tools

Manual audit

- Comparing the project to the Tomb Finance implementation

Classification of issue severity

High severity	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
Medium severity	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.
Low severity	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

Issues

High severity issues

No issues were found

Medium severity issues

No issues were found

Low severity issues

No issues were found

Conclusion

The EMP Money Project was compared with the Tomb Project. EMP Money has changed the implementation of Treasury contract.

The Token contract is not affected by the vulnerability that was discovered in the Tomb Project since the TAX collection functionality is never used in the deployed contract at address 0x3b248CEfA87F836a4e6f6d6c9b42991b88Dc1d58.

In the contract Treasury changed the array of pools excludedFromTotalSupply.

No serious issues were found in the audited changes.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability)set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without 0xGuard prior written consent.

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts 0xGuard to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Static code analysis results

```
Reentrancy in HSharesRewardPool.deposit(uint256,uint256)
(HSharesRewardPool.sol#757-775):
⊠External calls:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#765)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
⊠⊠- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
⊠⊠- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠- pool.token.safeTransferFrom(_sender,address(this),_amount)
(HSharesRewardPool.sol#770)
⊠External calls sending eth:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#765)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠State variables written after the call(s):
⊠- user.amount = user.amount.add(_amount) (HSharesRewardPool.sol#771)
⊠- user.rewardDebt = user.amount.mul(pool.accHSharesPerShare).div(1e18)
(HSharesRewardPool.sol#773)
Reentrancy in HSharesRewardPool.withdraw(uint256,uint256)
(HSharesRewardPool.sol#778-795):
⊠External calls:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
⊠⊠- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
⊠⊠- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠External calls sending eth:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠State variables written after the call(s):
⊠- user.amount = user.amount.sub(_amount) (HSharesRewardPool.sol#790)
Reentrancy in HSharesRewardPool.withdraw(uint256,uint256)
(HSharesRewardPool.sol#778-795):
⊠External calls:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
```

```
⊠⊠- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
⊠⊠- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠- pool.token.safeTransfer(_sender,_amount) (HSharesRewardPool.sol#791)
⊠External calls sending eth:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠State variables written after the call(s):
⊠- user.rewardDebt = user.amount.mul(pool.accHSharesPerShare).div(1e18)
(HSharesRewardPool.sol#793)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities

HSharesRewardPool.pendingShare(uint256,address) (HSharesRewardPool.sol#712-723)
performs a multiplication on the result of a division:
⊠-_bshareReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(HSharesRewardPool.sol#719)
⊠-accHSharesPerShare = accHSharesPerShare.add(_bshareReward.mul(1e18).div(tokenSupply))
(HSharesRewardPool.sol#720)
HSharesRewardPool.updatePool(uint256) (HSharesRewardPool.sol#734-754) performs a
multiplication on the result of a division:
⊠-_bshareReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(HSharesRewardPool.sol#750)
⊠-pool.accHSharesPerShare =
pool.accHSharesPerShare.add(_bshareReward.mul(1e18).div(tokenSupply))
(HSharesRewardPool.sol#751)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply

HSharesRewardPool.updatePool(uint256) (HSharesRewardPool.sol#734-754) uses a dangerous
strict equality:
⊠- tokenSupply == 0 (HSharesRewardPool.sol#740)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-
strict-equalities

HSharesRewardPool.setOperator(address) (HSharesRewardPool.sol#820-822) should emit an
event for:
⊠- operator = _operator (HSharesRewardPool.sol#821)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
access-control
```

```
HSharesRewardPool.add(uint256,IERC20,bool,uint256) (HSharesRewardPool.sol#645-683)
should emit an event for:
⊠- totalAllocPoint = totalAllocPoint.add(_allocPoint) (HSharesRewardPool.sol#681)
HSharesRewardPool.set(uint256,uint256) (HSharesRewardPool.sol#686-695) should emit an
event for:
⊠- totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint)
(HSharesRewardPool.sol#690-692)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

HSharesRewardPool.setOperator(address)._operator (HSharesRewardPool.sol#820) lacks a
zero-check on :
⊠⊠- operator = _operator (HSharesRewardPool.sol#821)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

Reentrancy in HSharesRewardPool.deposit(uint256,uint256)
(HSharesRewardPool.sol#757-775):
⊠External calls:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#765)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
⊠⊠- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
⊠⊠- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠External calls sending eth:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#765)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠Event emitted after the call(s):
⊠- RewardPaid(_sender,_pending) (HSharesRewardPool.sol#766)
Reentrancy in HSharesRewardPool.deposit(uint256,uint256)
(HSharesRewardPool.sol#757-775):
⊠External calls:
⊠- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#765)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
⊠⊠- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
⊠⊠- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
⊠⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
⊠- pool.token.safeTransferFrom(_sender,address(this),_amount)
(HSharesRewardPool.sol#770)
```

```
	External calls sending eth:
	- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#765)
		- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
	Event emitted after the call(s):
	- Deposit(_sender,_pid,_amount) (HSharesRewardPool.sol#774)
Reentrancy in HSharesRewardPool.emergencyWithdraw(uint256)
(HSharesRewardPool.sol#798-806):
	External calls:
	- pool.token.safeTransfer(msg.sender,_amount) (HSharesRewardPool.sol#804)
	Event emitted after the call(s):
	- EmergencyWithdraw(msg.sender,_pid,_amount) (HSharesRewardPool.sol#805)
Reentrancy in HSharesRewardPool.withdraw(uint256,uint256)
(HSharesRewardPool.sol#778-795):
	External calls:
	- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
		- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
		- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
		- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
	External calls sending eth:
	- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
		- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
	Event emitted after the call(s):
	- RewardPaid(_sender,_pending) (HSharesRewardPool.sol#787)
Reentrancy in HSharesRewardPool.withdraw(uint256,uint256)
(HSharesRewardPool.sol#778-795):
	External calls:
	- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(HSharesRewardPool.sol#556)
		- bshare.safeTransfer(_to,_bshareBal) (HSharesRewardPool.sol#813)
		- bshare.safeTransfer(_to,_amount) (HSharesRewardPool.sol#815)
		- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
	- pool.token.safeTransfer(_sender,_amount) (HSharesRewardPool.sol#791)
	External calls sending eth:
	- safeHSharesTransfer(_sender,_pending) (HSharesRewardPool.sol#786)
		- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
	Event emitted after the call(s):
	- Withdraw(_sender,_pid,_amount) (HSharesRewardPool.sol#794)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3
```

```
HSharesRewardPool.constructor(address,uint256) (HSharesRewardPool.sol#621-630) uses
timestamp for comparisons
⊠Dangerous comparisons:
⊠- require(bool,string)(block.timestamp < _poolStartTime,late)
(HSharesRewardPool.sol#625)
HSharesRewardPool.checkPoolDuplicate(IERC20) (HSharesRewardPool.sol#637-642) uses
timestamp for comparisons
⊠Dangerous comparisons:
⊠- pid < length (HSharesRewardPool.sol#639)
⊠- require(bool,string)(poolInfo[pid].token != _token,HSharesRewardPool: existing
pool?) (HSharesRewardPool.sol#640)
HSharesRewardPool.add(uint256,IERC20,bool,uint256) (HSharesRewardPool.sol#645-683) uses
timestamp for comparisons
⊠Dangerous comparisons:
⊠- block.timestamp < poolStartTime (HSharesRewardPool.sol#655)
⊠- _lastRewardTime == 0 (HSharesRewardPool.sol#657)
⊠- _lastRewardTime < poolStartTime (HSharesRewardPool.sol#660)
⊠- _lastRewardTime == 0 || _lastRewardTime < block.timestamp
(HSharesRewardPool.sol#666)
⊠- _isStarted = (_lastRewardTime <= poolStartTime) || (_lastRewardTime <=
block.timestamp) (HSharesRewardPool.sol#670-672)
HSharesRewardPool.getGeneratedReward(uint256,uint256) (HSharesRewardPool.sol#698-709)
uses timestamp for comparisons
⊠Dangerous comparisons:
⊠- _fromTime >= _toTime (HSharesRewardPool.sol#699)
⊠- _toTime >= poolEndTime (HSharesRewardPool.sol#700)
⊠- _toTime <= poolStartTime (HSharesRewardPool.sol#705)
HSharesRewardPool.pendingShare(uint256,address) (HSharesRewardPool.sol#712-723) uses
timestamp for comparisons
⊠Dangerous comparisons:
⊠- block.timestamp > pool.lastRewardTime && tokenSupply != 0
(HSharesRewardPool.sol#717)
HSharesRewardPool.massUpdatePools() (HSharesRewardPool.sol#726-731) uses timestamp for
comparisons
⊠Dangerous comparisons:
⊠- pid < length (HSharesRewardPool.sol#728)
HSharesRewardPool.updatePool(uint256) (HSharesRewardPool.sol#734-754) uses timestamp
for comparisons
⊠Dangerous comparisons:
⊠- block.timestamp <= pool.lastRewardTime (HSharesRewardPool.sol#736)
```

```
HSharesRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)
(HSharesRewardPool.sol#824-835) uses timestamp for comparisons
⊠Dangerous comparisons:
⊠- block.timestamp < poolEndTime + 7776000 (HSharesRewardPool.sol#825)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp

Address.isContract(address) (HSharesRewardPool.sol#324-333) uses assembly
⊠- INLINE ASM (HSharesRewardPool.sol#331)
Address._verifyCallResult(bool,bytes,string) (HSharesRewardPool.sol#469-486) uses
assembly
⊠- INLINE ASM (HSharesRewardPool.sol#478-481)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
⊠- Version used: ['0.6.12', '>=0.6.0<0.8.0', '>=0.6.2<0.8.0']
⊠- >=0.6.0<0.8.0 (HSharesRewardPool.sol#6)
⊠- >=0.6.0<0.8.0 (HSharesRewardPool.sol#85)
⊠- >=0.6.2<0.8.0 (HSharesRewardPool.sol#301)
⊠- >=0.6.0<0.8.0 (HSharesRewardPool.sol#492)
⊠- 0.6.12 (HSharesRewardPool.sol#567)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used

Address.functionCall(address,bytes) (HSharesRewardPool.sol#377-379) is never used and
should be removed
Address.functionCallWithValue(address,bytes,uint256) (HSharesRewardPool.sol#402-404) is
never used and should be removed
Address.functionDelegateCall(address,bytes) (HSharesRewardPool.sol#451-453) is never
used and should be removed
Address.functionDelegateCall(address,bytes,string) (HSharesRewardPool.sol#461-467) is
never used and should be removed
Address.functionStaticCall(address,bytes) (HSharesRewardPool.sol#427-429) is never used
and should be removed
Address.functionStaticCall(address,bytes,string) (HSharesRewardPool.sol#437-443) is
never used and should be removed
Address.sendValue(address,uint256) (HSharesRewardPool.sol#351-357) is never used and
should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (HSharesRewardPool.sol#524-533) is never
used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (HSharesRewardPool.sol#540-543)
```

```
is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (HSharesRewardPool.sol#535-538)
is never used and should be removed
SafeMath.div(uint256,uint256,string) (HSharesRewardPool.sol#272-275) is never used and
should be removed
SafeMath.mod(uint256,uint256) (HSharesRewardPool.sol#234-237) is never used and should
be removed
SafeMath.mod(uint256,uint256,string) (HSharesRewardPool.sol#292-295) is never used and
should be removed
SafeMath.sub(uint256,uint256,string) (HSharesRewardPool.sol#252-255) is never used and
should be removed
SafeMath.tryAdd(uint256,uint256) (HSharesRewardPool.sol#106-110) is never used and
should be removed
SafeMath.tryDiv(uint256,uint256) (HSharesRewardPool.sol#142-145) is never used and
should be removed
SafeMath.tryMod(uint256,uint256) (HSharesRewardPool.sol#152-155) is never used and
should be removed
SafeMath.tryMul(uint256,uint256) (HSharesRewardPool.sol#127-135) is never used and
should be removed
SafeMath.trySub(uint256,uint256) (HSharesRewardPool.sol#117-120) is never used and
should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.6.0<0.8.0 (HSharesRewardPool.sol#6) is too complex
Pragma version>=0.6.0<0.8.0 (HSharesRewardPool.sol#85) is too complex
Pragma version>=0.6.2<0.8.0 (HSharesRewardPool.sol#301) is too complex
Pragma version>=0.6.0<0.8.0 (HSharesRewardPool.sol#492) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Low level call in Address.sendValue(address,uint256) (HSharesRewardPool.sol#351-357):
⊠- (success) = recipient.call{value: amount}() (HSharesRewardPool.sol#355)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string)
(HSharesRewardPool.sol#412-419):
⊠- (success,returndata) = target.call{value: value}(data) (HSharesRewardPool.sol#417)
Low level call in Address.functionStaticCall(address,bytes,string)
(HSharesRewardPool.sol#437-443):
⊠- (success,returndata) = target.staticcall(data) (HSharesRewardPool.sol#441)
Low level call in Address.functionDelegateCall(address,bytes,string)
(HSharesRewardPool.sol#461-467):
⊠- (success,returndata) = target.delegatecall(data) (HSharesRewardPool.sol#465)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-
calls

Parameter HSharesRewardPool.checkPoolDuplicate(IERC20)._token
(HSharesRewardPool.sol#637) is not in mixedCase
Parameter HSharesRewardPool.add(uint256,IERC20,bool,uint256)._allocPoint
(HSharesRewardPool.sol#646) is not in mixedCase
Parameter HSharesRewardPool.add(uint256,IERC20,bool,uint256)._token
(HSharesRewardPool.sol#647) is not in mixedCase
Parameter HSharesRewardPool.add(uint256,IERC20,bool,uint256)._withUpdate
(HSharesRewardPool.sol#648) is not in mixedCase
Parameter HSharesRewardPool.add(uint256,IERC20,bool,uint256)._lastRewardTime
(HSharesRewardPool.sol#649) is not in mixedCase
Parameter HSharesRewardPool.set(uint256,uint256)._pid (HSharesRewardPool.sol#686) is
not in mixedCase
Parameter HSharesRewardPool.set(uint256,uint256)._allocPoint
(HSharesRewardPool.sol#686) is not in mixedCase
Parameter HSharesRewardPool.getGeneratedReward(uint256,uint256)._fromTime
(HSharesRewardPool.sol#698) is not in mixedCase
Parameter HSharesRewardPool.getGeneratedReward(uint256,uint256)._toTime
(HSharesRewardPool.sol#698) is not in mixedCase
Parameter HSharesRewardPool.pendingShare(uint256,address)._pid
(HSharesRewardPool.sol#712) is not in mixedCase
Parameter HSharesRewardPool.pendingShare(uint256,address)._user
(HSharesRewardPool.sol#712) is not in mixedCase
Parameter HSharesRewardPool.updatePool(uint256)._pid (HSharesRewardPool.sol#734) is not
in mixedCase
Parameter HSharesRewardPool.deposit(uint256,uint256)._pid (HSharesRewardPool.sol#757)
is not in mixedCase
Parameter HSharesRewardPool.deposit(uint256,uint256)._amount
(HSharesRewardPool.sol#757) is not in mixedCase
Parameter HSharesRewardPool.withdraw(uint256,uint256)._pid (HSharesRewardPool.sol#778)
is not in mixedCase
Parameter HSharesRewardPool.withdraw(uint256,uint256)._amount
(HSharesRewardPool.sol#778) is not in mixedCase
Parameter HSharesRewardPool.emergencyWithdraw(uint256)._pid (HSharesRewardPool.sol#798)
is not in mixedCase
Parameter HSharesRewardPool.safeHSharesTransfer(address,uint256)._to
(HSharesRewardPool.sol#809) is not in mixedCase
Parameter HSharesRewardPool.safeHSharesTransfer(address,uint256)._amount
(HSharesRewardPool.sol#809) is not in mixedCase
```

```
Parameter HSharesRewardPool.setOperator(address)._operator (HSharesRewardPool.sol#820)
is not in mixedCase
Parameter HSharesRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token
(HSharesRewardPool.sol#824) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

HSharesRewardPool.runningTime (HSharesRewardPool.sol#613) should be constant
HSharesRewardPool.tSharePerSecond (HSharesRewardPool.sol#612) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-
variables-that-could-be-declared-constant

set(uint256,uint256) should be declared external:
⊠- HSharesRewardPool.set(uint256,uint256) (HSharesRewardPool.sol#686-695)
deposit(uint256,uint256) should be declared external:
⊠- HSharesRewardPool.deposit(uint256,uint256) (HSharesRewardPool.sol#757-775)
withdraw(uint256,uint256) should be declared external:
⊠- HSharesRewardPool.withdraw(uint256,uint256) (HSharesRewardPool.sol#778-795)
emergencyWithdraw(uint256) should be declared external:
⊠- HSharesRewardPool.emergencyWithdraw(uint256) (HSharesRewardPool.sol#798-806)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external

HShares.governanceRecoverUnsupported(IERC20,uint256,address) (HShares.sol#925-931)
ignores return value by _token.transfer(_to,_amount) (HShares.sol#930)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

HShares.setTreasuryFund(address)._communityFund (HShares.sol#852) lacks a zero-check
on :
⊠⊠- communityFund = _communityFund (HShares.sol#854)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

HShares.unclaimedTreasuryFund() (HShares.sol#869-874) uses timestamp for comparisons
⊠Dangerous comparisons:
⊠- _now > endTime (HShares.sol#871)
⊠- communityFundLastClaimed >= _now (HShares.sol#872)
HShares.unclaimedDevFund() (HShares.sol#876-881) uses timestamp for comparisons
⊠Dangerous comparisons:
⊠- _now > endTime (HShares.sol#878)
```

```
- devFundLastClaimed >= _now (HShares.sol#879)
HShares.unclaimedTeam1Fund() (HShares.sol#883-888) uses timestamp for comparisons
Dangerous comparisons:
- _now > endTime (HShares.sol#885)
- team1FundLastClaimed >= _now (HShares.sol#886)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp

Different versions of Solidity is used:
- Version used: ['0.6.12', '>=0.6.0<0.8.0']
- >=0.6.0<0.8.0 (HShares.sol#6)
- >=0.6.0<0.8.0 (HShares.sol#222)
- >=0.6.0<0.8.0 (HShares.sol#248)
- >=0.6.0<0.8.0 (HShares.sol#327)
- >=0.6.0<0.8.0 (HShares.sol#633)
- >=0.6.0<0.8.0 (HShares.sol#675)
- >=0.6.0<0.8.0 (HShares.sol#680)
- 0.6.12 (HShares.sol#749)
- 0.6.12 (HShares.sol#789)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used

Context._msgData() (HShares.sol#239-242) is never used and should be removed
ERC20._setupDecimals(uint8) (HShares.sol#609-611) is never used and should be removed
SafeMath.div(uint256,uint256,string) (HShares.sol#193-196) is never used and should be
removed
SafeMath.mod(uint256,uint256) (HShares.sol#155-158) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (HShares.sol#213-216) is never used and should be
removed
SafeMath.tryAdd(uint256,uint256) (HShares.sol#27-31) is never used and should be
removed
SafeMath.tryDiv(uint256,uint256) (HShares.sol#63-66) is never used and should be
removed
SafeMath.tryMod(uint256,uint256) (HShares.sol#73-76) is never used and should be
removed
SafeMath.tryMul(uint256,uint256) (HShares.sol#48-56) is never used and should be
removed
SafeMath.trySub(uint256,uint256) (HShares.sol#38-41) is never used and should be
removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
Pragma version>=0.6.0<0.8.0 (HShares.sol#6) is too complex
Pragma version>=0.6.0<0.8.0 (HShares.sol#222) is too complex
Pragma version>=0.6.0<0.8.0 (HShares.sol#248) is too complex
Pragma version>=0.6.0<0.8.0 (HShares.sol#327) is too complex
Pragma version>=0.6.0<0.8.0 (HShares.sol#633) is too complex
Pragma version>=0.6.0<0.8.0 (HShares.sol#675) is too complex
Pragma version>=0.6.0<0.8.0 (HShares.sol#680) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Parameter HShares.setTreasuryFund(address)._communityFund (HShares.sol#852) is not in
mixedCase
Parameter HShares.setDevFund(address)._devFund (HShares.sol#857) is not in mixedCase
Parameter HShares.setTeam1Fund(address)._team1Fund (HShares.sol#863) is not in
mixedCase
Parameter HShares.distributeReward(address)._farmingIncentiveFund (HShares.sol#914) is
not in mixedCase
Parameter HShares.governanceRecoverUnsupported(IERC20,uint256,address)._token
(HShares.sol#926) is not in mixedCase
Parameter HShares.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(HShares.sol#927) is not in mixedCase
Parameter HShares.governanceRecoverUnsupported(IERC20,uint256,address)._to
(HShares.sol#928) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (HShares.sol#240)" inContext (HShares.sol#234-243)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

name() should be declared external:
- ERC20.name() (HShares.sol#386-388)
symbol() should be declared external:
- ERC20.symbol() (HShares.sol#394-396)
decimals() should be declared external:
- ERC20.decimals() (HShares.sol#411-413)
totalSupply() should be declared external:
- ERC20.totalSupply() (HShares.sol#418-420)
balanceOf(address) should be declared external:
- ERC20.balanceOf(address) (HShares.sol#425-427)
transfer(address,uint256) should be declared external:
```

```
- ERC20.transfer(address,uint256) (HShares.sol#437-440)
approve(address,uint256) should be declared external:
- ERC20.approve(address,uint256) (HShares.sol#456-459)
transferFrom(address,address,uint256) should be declared external:
- ERC20.transferFrom(address,address,uint256) (HShares.sol#474-478)
increaseAllowance(address,uint256) should be declared external:
- ERC20.increaseAllowance(address,uint256) (HShares.sol#492-495)
decreaseAllowance(address,uint256) should be declared external:
- ERC20.decreaseAllowance(address,uint256) (HShares.sol#511-514)
burnFrom(address,uint256) should be declared external:
- ERC20Burnable.burnFrom(address,uint256) (HShares.sol#664-669)
renounceOwnership() should be declared external:
- Ownable.renounceOwnership() (HShares.sol#730-733)
transferOwnership(address) should be declared external:
- Ownable.transferOwnership(address) (HShares.sol#739-743)
operator() should be declared external:
- Operator.operator() (HShares.sol#762-764)
isOperator() should be declared external:
- Operator.isOperator() (HShares.sol#771-773)
transferOperator(address) should be declared external:
- Operator.transferOperator(address) (HShares.sol#775-777)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
```

```
Hermes.governanceRecoverUnsupported(IERC20,uint256,address) (Hermes.sol#1253-1259)
ignores return value by _token.transfer(_to,_amount) (Hermes.sol#1258)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

Hermes.setTaxTiersTwap(uint8,uint256) (Hermes.sol#1084-1095) contains a tautology or
contradiction:
- require(bool,string)(_index >= 0,Index has to be higher than 0) (Hermes.sol#1085)
Hermes.setTaxTiersRate(uint8,uint256) (Hermes.sol#1097-1102) contains a tautology or
contradiction:
- require(bool,string)(_index >= 0,Index has to be higher than 0) (Hermes.sol#1098)
Hermes._updateTaxRate(uint256) (Hermes.sol#1116-1126) contains a tautology or
contradiction:
- tierId >= 0 (Hermes.sol#1118)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#tautology-or-
contradiction
```

```
Hermes._getHermesPrice()._price (Hermes.sol#1109) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-
local-variables

Hermes._getHermesPrice() (Hermes.sol#1108-1114) ignores return value by
IOracle(hermesOracle).consult(address(this),1e18) (Hermes.sol#1109-1113)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Hermes.setBurnThreshold(uint256) (Hermes.sol#1104-1106) should emit an event for:
⊠- burnThreshold = _burnThreshold (Hermes.sol#1105)
Hermes.setTaxRate(uint256) (Hermes.sol#1152-1156) should emit an event for:
⊠- taxRate = _taxRate (Hermes.sol#1155)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

Variable 'Hermes._getHermesPrice()._price (Hermes.sol#1109)' in
Hermes._getHermesPrice() (Hermes.sol#1108-1114) potentially used before declaration:
uint256(_price) (Hermes.sol#1110)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-
declaration-usage-of-local-variables

Different versions of Solidity is used:
⊠- Version used: ['0.6.12', '>=0.6.0<0.8.0']
⊠- >=0.6.0<0.8.0 (Hermes.sol#6)
⊠- >=0.6.0<0.8.0 (Hermes.sol#32)
⊠- >=0.6.0<0.8.0 (Hermes.sol#111)
⊠- >=0.6.0<0.8.0 (Hermes.sol#327)
⊠- >=0.6.0<0.8.0 (Hermes.sol#633)
⊠- >=0.6.0<0.8.0 (Hermes.sol#675)
⊠- 0.6.12 (Hermes.sol#708)
⊠- >=0.6.0<0.8.0 (Hermes.sol#869)
⊠- >=0.6.0<0.8.0 (Hermes.sol#874)
⊠- 0.6.12 (Hermes.sol#943)
⊠- 0.6.12 (Hermes.sol#983)
⊠- 0.6.12 (Hermes.sol#996)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used

Hermes._updateTaxRate(uint256) (Hermes.sol#1116-1126) has costly operations inside a
loop:
```

```
- taxRate = taxTiersRates[tierId] (Hermes.sol#1121)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-
operations-inside-a-loop

Context._msgData() (Hermes.sol#23-26) is never used and should be removed
ERC20._setupDecimals(uint8) (Hermes.sol#609-611) is never used and should be removed
Math.average(uint256,uint256) (Hermes.sol#699-702) is never used and should be removed
Math.max(uint256,uint256) (Hermes.sol#684-686) is never used and should be removed
Math.min(uint256,uint256) (Hermes.sol#691-693) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Hermes.sol#298-301) is never used and should be
removed
SafeMath.mod(uint256,uint256) (Hermes.sol#260-263) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Hermes.sol#318-321) is never used and should be
removed
SafeMath.tryAdd(uint256,uint256) (Hermes.sol#132-136) is never used and should be
removed
SafeMath.tryDiv(uint256,uint256) (Hermes.sol#168-171) is never used and should be
removed
SafeMath.tryMod(uint256,uint256) (Hermes.sol#178-181) is never used and should be
removed
SafeMath.tryMul(uint256,uint256) (Hermes.sol#153-161) is never used and should be
removed
SafeMath.trySub(uint256,uint256) (Hermes.sol#143-146) is never used and should be
removed
SafeMath8.add(uint8,uint8) (Hermes.sol#734-739) is never used and should be removed
SafeMath8.div(uint8,uint8) (Hermes.sol#808-810) is never used and should be removed
SafeMath8.div(uint8,uint8,string) (Hermes.sol#824-830) is never used and should be
removed
SafeMath8.mod(uint8,uint8) (Hermes.sol#844-846) is never used and should be removed
SafeMath8.mod(uint8,uint8,string) (Hermes.sol#860-863) is never used and should be
removed
SafeMath8.mul(uint8,uint8) (Hermes.sol#782-794) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.6.0<0.8.0 (Hermes.sol#6) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#32) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#111) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#327) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#633) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#675) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#869) is too complex
Pragma version>=0.6.0<0.8.0 (Hermes.sol#874) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Parameter Hermes.isAddressExcluded(address)._address (Hermes.sol#1080) is not in
mixedCase
Parameter Hermes.setTaxTiersTwap(uint8,uint256)._index (Hermes.sol#1084) is not in
mixedCase
Parameter Hermes.setTaxTiersTwap(uint8,uint256)._value (Hermes.sol#1084) is not in
mixedCase
Parameter Hermes.setTaxTiersRate(uint8,uint256)._index (Hermes.sol#1097) is not in
mixedCase
Parameter Hermes.setTaxTiersRate(uint8,uint256)._value (Hermes.sol#1097) is not in
mixedCase
Parameter Hermes.setBurnThreshold(uint256)._burnThreshold (Hermes.sol#1104) is not in
mixedCase
Parameter Hermes.setHermesOracle(address)._hermesOracle (Hermes.sol#1136) is not in
mixedCase
Parameter Hermes.setTaxOffice(address)._taxOffice (Hermes.sol#1141) is not in mixedCase
Parameter Hermes.setTaxCollectorAddress(address)._taxCollectorAddress (Hermes.sol#1147)
is not in mixedCase
Parameter Hermes.setTaxRate(uint256)._taxRate (Hermes.sol#1152) is not in mixedCase
Parameter Hermes.excludeAddress(address)._address (Hermes.sol#1158) is not in mixedCase
Parameter Hermes.includeAddress(address)._address (Hermes.sol#1164) is not in mixedCase
Parameter Hermes.distributeReward(address)._launcherAddress (Hermes.sol#1245) is not in
mixedCase
Parameter Hermes.governanceRecoverUnsupported(IERC20,uint256,address)._token
(Hermes.sol#1254) is not in mixedCase
Parameter Hermes.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(Hermes.sol#1255) is not in mixedCase
Parameter Hermes.governanceRecoverUnsupported(IERC20,uint256,address)._to
(Hermes.sol#1256) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (Hermes.sol#24)" inContext (Hermes.sol#18-27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

name() should be declared external:
	- ERC20.name() (Hermes.sol#386-388)
```

```
symbol() should be declared external:
⊠- ERC20.symbol() (Hermes.sol#394-396)
decimals() should be declared external:
⊠- ERC20.decimals() (Hermes.sol#411-413)
totalSupply() should be declared external:
⊠- ERC20.totalSupply() (Hermes.sol#418-420)
transfer(address,uint256) should be declared external:
⊠- ERC20.transfer(address,uint256) (Hermes.sol#437-440)
approve(address,uint256) should be declared external:
⊠- ERC20.approve(address,uint256) (Hermes.sol#456-459)
transferFrom(address,address,uint256) should be declared external:
⊠- ERC20.transferFrom(address,address,uint256) (Hermes.sol#474-478)
⊠- Hermes.transferFrom(address,address,uint256) (Hermes.sol#1192-1216)
increaseAllowance(address,uint256) should be declared external:
⊠- ERC20.increaseAllowance(address,uint256) (Hermes.sol#492-495)
decreaseAllowance(address,uint256) should be declared external:
⊠- ERC20.decreaseAllowance(address,uint256) (Hermes.sol#511-514)
renounceOwnership() should be declared external:
⊠- Ownable.renounceOwnership() (Hermes.sol#924-927)
transferOwnership(address) should be declared external:
⊠- Ownable.transferOwnership(address) (Hermes.sol#933-937)
operator() should be declared external:
⊠- Operator.operator() (Hermes.sol#956-958)
transferOperator(address) should be declared external:
⊠- Operator.transferOperator(address) (Hermes.sol#969-971)
isAddressExcluded(address) should be declared external:
⊠- Hermes.isAddressExcluded(address) (Hermes.sol#1080-1082)
setTaxTiersTwap(uint8,uint256) should be declared external:
⊠- Hermes.setTaxTiersTwap(uint8,uint256) (Hermes.sol#1084-1095)
setTaxTiersRate(uint8,uint256) should be declared external:
⊠- Hermes.setTaxTiersRate(uint8,uint256) (Hermes.sol#1097-1102)
setBurnThreshold(uint256) should be declared external:
⊠- Hermes.setBurnThreshold(uint256) (Hermes.sol#1104-1106)
enableAutoCalculateTax() should be declared external:
⊠- Hermes.enableAutoCalculateTax() (Hermes.sol#1128-1130)
disableAutoCalculateTax() should be declared external:
⊠- Hermes.disableAutoCalculateTax() (Hermes.sol#1132-1134)
setHermesOracle(address) should be declared external:
⊠- Hermes.setHermesOracle(address) (Hermes.sol#1136-1139)
setTaxOffice(address) should be declared external:
⊠- Hermes.setTaxOffice(address) (Hermes.sol#1141-1145)
```

```
setTaxCollectorAddress(address) should be declared external:
	- Hermes.setTaxCollectorAddress(address) (Hermes.sol#1147-1150)
setTaxRate(uint256) should be declared external:
	- Hermes.setTaxRate(uint256) (Hermes.sol#1152-1156)
includeAddress(address) should be declared external:
	- Hermes.includeAddress(address) (Hermes.sol#1164-1168)
mint(address,uint256) should be declared external:
	- Hermes.mint(address,uint256) (Hermes.sol#1176-1182)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
```

```
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1427-1467):
	External calls:
	- _updateHermesPrice() (Treasury.sol#1428)
		- IOracle(hermesOracle).update() (Treasury.sol#1320)
	- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#589)
		- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
		- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
		- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
		- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
		- IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
		- IERC20(hermes).safeApprove(olympus,0) (Treasury.sol#1411)
		- IERC20(hermes).safeApprove(olympus,_amount) (Treasury.sol#1412)
		- IOlympus(olympus).allocateSeigniorage(_amount) (Treasury.sol#1413)
	External calls sending eth:
	- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
		- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
	State variables written after the call(s):
	- seigniorageSaved = seigniorageSaved.add(_savedForBond) (Treasury.sol#1461)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities

Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415) ignores return value by
IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415) ignores return value by
IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415) ignores return value by
IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

Treasury.allocateSeigniorage() (Treasury.sol#1427-1467) performs a multiplication on
the result of a division:
⊠-_seigniorage = hermesSupply.mul(_percentage).div(1e18) (Treasury.sol#1450)
⊠-_savedForOlympus = _seigniorage.mul(seigniorageExpansionFloorPercent).div(10000)
(Treasury.sol#1451)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply

Reentrancy in Treasury.buyBonds(uint256,uint256) (Treasury.sol#1330-1357):
⊠External calls:
⊠- IBasisAsset(hermes).burnFrom(msg.sender,_hermesAmount) (Treasury.sol#1350)
⊠- IBasisAsset(bbond).mint(msg.sender,_bondAmount) (Treasury.sol#1351)
⊠State variables written after the call(s):
⊠- epochSupplyContractionLeft = epochSupplyContractionLeft.sub(_hermesAmount)
(Treasury.sol#1353)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-1

Treasury.setSupplyTiersEntry(uint8,uint256) (Treasury.sol#1223-1234) contains a
tautology or contradiction:
⊠- require(bool,string)(_index >= 0,Index has to be higher than 0) (Treasury.sol#1224)
Treasury.setMaxExpansionTiersEntry(uint8,uint256) (Treasury.sol#1236-1242) contains a
tautology or contradiction:
⊠- require(bool,string)(_index >= 0,Index has to be higher than 0) (Treasury.sol#1237)
Treasury._calculateMaxSupplyExpansionPercent(uint256) (Treasury.sol#1417-1425) contains
a tautology or contradiction:
⊠- tierId >= 0 (Treasury.sol#1418)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#tautology-or-
contradiction

Treasury.allocateSeigniorage()._savedForBond (Treasury.sol#1439) is a local variable
never initialized
Treasury.getHermesUpdatedPrice().price (Treasury.sol#1083) is a local variable never
initialized
Treasury.getHermesPrice().price (Treasury.sol#1075) is a local variable never
initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-
local-variables
```

```
Treasury.getHermesPrice() (Treasury.sol#1074-1080) ignores return value by
IOracle(hermesOracle).consult(hermes,1e18) (Treasury.sol#1075-1079)
Treasury.getHermesUpdatedPrice() (Treasury.sol#1082-1088) ignores return value by
IOracle(hermesOracle).twap(hermes,1e18) (Treasury.sol#1083-1087)
Treasury.buyBonds(uint256,uint256) (Treasury.sol#1330-1357) ignores return value by
IBasisAsset(bbond).mint(msg.sender,_bondAmount) (Treasury.sol#1351)
Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415) ignores return value by
IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
Treasury.allocateSeigniorage() (Treasury.sol#1427-1467) ignores return value by
IBasisAsset(hermes).mint(address(this),_savedForBond) (Treasury.sol#1462)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Treasury.setOperator(address) (Treasury.sol#1201-1203) should emit an event for:
⊠- operator = _operator (Treasury.sol#1202)
Treasury.setOlympus(address) (Treasury.sol#1205-1207) should emit an event for:
⊠- olympus = _olympus (Treasury.sol#1206)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
access-control

Treasury.setHermesPriceCeiling(uint256) (Treasury.sol#1213-1216) should emit an event
for:
⊠- hermesPriceCeiling = _hermesPriceCeiling (Treasury.sol#1215)
Treasury.setMaxSupplyExpansionPercents(uint256) (Treasury.sol#1218-1221) should emit an
event for:
⊠- maxSupplyExpansionPercent = _maxSupplyExpansionPercent (Treasury.sol#1220)
Treasury.setBondDepletionFloorPercent(uint256) (Treasury.sol#1244-1247) should emit an
event for:
⊠- bondDepletionFloorPercent = _bondDepletionFloorPercent (Treasury.sol#1246)
Treasury.setMaxDebtRatioPercent(uint256) (Treasury.sol#1254-1257) should emit an event
for:
⊠- maxDebtRatioPercent = _maxDebtRatioPercent (Treasury.sol#1256)
Treasury.setBootstrap(uint256,uint256) (Treasury.sol#1259-1264) should emit an event
for:
⊠- bootstrapEpochs = _bootstrapEpochs (Treasury.sol#1262)
⊠- bootstrapSupplyExpansionPercent = _bootstrapSupplyExpansionPercent
(Treasury.sol#1263)
Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)
(Treasury.sol#1266-1286) should emit an event for:
⊠- daoFundSharedPercent = _daoFundSharedPercent (Treasury.sol#1281)
⊠- devFundSharedPercent = _devFundSharedPercent (Treasury.sol#1283)
```

```
- team1FundSharedPercent = _team1FundSharedPercent (Treasury.sol#1285)
Treasury.setMaxDiscountRate(uint256) (Treasury.sol#1288-1290) should emit an event for:
- maxDiscountRate = _maxDiscountRate (Treasury.sol#1289)
Treasury.setMaxPremiumRate(uint256) (Treasury.sol#1292-1294) should emit an event for:
- maxPremiumRate = _maxPremiumRate (Treasury.sol#1293)
Treasury.setDiscountPercent(uint256) (Treasury.sol#1296-1299) should emit an event for:
- discountPercent = _discountPercent (Treasury.sol#1298)
Treasury.setPremiumThreshold(uint256) (Treasury.sol#1301-1305) should emit an event
for:
- premiumThreshold = _premiumThreshold (Treasury.sol#1304)
Treasury.setPremiumPercent(uint256) (Treasury.sol#1307-1310) should emit an event for:
- premiumPercent = _premiumPercent (Treasury.sol#1309)
Treasury.setMintingFactorForPayingDebt(uint256) (Treasury.sol#1312-1315) should emit an
event for:
- mintingFactorForPayingDebt = _mintingFactorForPayingDebt (Treasury.sol#1314)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

Treasury.initialize(address,address,address,address,address,uint256)._hermes
(Treasury.sol#1158) lacks a zero-check on :
 - hermes = _hermes (Treasury.sol#1165)
Treasury.initialize(address,address,address,address,address,uint256)._bbond
(Treasury.sol#1159) lacks a zero-check on :
 - bbond = _bbond (Treasury.sol#1166)
Treasury.initialize(address,address,address,address,address,uint256)._bshare
(Treasury.sol#1160) lacks a zero-check on :
 - bshare = _bshare (Treasury.sol#1167)
Treasury.initialize(address,address,address,address,address,uint256)._hermesOracle
(Treasury.sol#1161) lacks a zero-check on :
 - hermesOracle = _hermesOracle (Treasury.sol#1168)
Treasury.initialize(address,address,address,address,address,uint256)._olympus
(Treasury.sol#1162) lacks a zero-check on :
 - olympus = _olympus (Treasury.sol#1169)
Treasury.setOperator(address)._operator (Treasury.sol#1201) lacks a zero-check on :
 - operator = _operator (Treasury.sol#1202)
Treasury.setOlympus(address)._olympus (Treasury.sol#1205) lacks a zero-check on :
 - olympus = _olympus (Treasury.sol#1206)
Treasury.setHermesOracle(address)._hermesOracle (Treasury.sol#1209) lacks a zero-check
on :
 - hermesOracle = _hermesOracle (Treasury.sol#1210)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
```

```
address-validation

Variable 'Treasury.getHermesPrice().price (Treasury.sol#1075)' in
Treasury.getHermesPrice() (Treasury.sol#1074-1080) potentially used before declaration:
uint256(price) (Treasury.sol#1076)
Variable 'Treasury.getHermesUpdatedPrice().price (Treasury.sol#1083)' in
Treasury.getHermesUpdatedPrice() (Treasury.sol#1082-1088) potentially used before
declaration: uint256(price) (Treasury.sol#1084)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-
declaration-usage-of-local-variables

Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1427-1467):
⊠External calls:
⊠- _updateHermesPrice() (Treasury.sol#1428)
⊠⊠- IOracle(hermesOracle).update() (Treasury.sol#1320)
⊠State variables written after the call(s):
⊠- _mse = _calculateMaxSupplyExpansionPercent(hermesSupply).mul(1e14)
(Treasury.sol#1441)
⊠⊠- maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1420)
⊠- previousEpochHermesPrice = getHermesPrice() (Treasury.sol#1429)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-2

Reentrancy in Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415):
⊠External calls:
⊠- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
⊠- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
⊠Event emitted after the call(s):
⊠- DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1392)
Reentrancy in Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415):
⊠External calls:
⊠- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
⊠- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
⊠- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
⊠Event emitted after the call(s):
⊠- DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1399)
Reentrancy in Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415):
⊠External calls:
⊠- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
⊠- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
⊠- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
```

```
- IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
Event emitted after the call(s):
- TeamFundFunded(now,_team1FundSharedAmount) (Treasury.sol#1406)
Reentrancy in Treasury._sendToOlympus(uint256) (Treasury.sol#1385-1415):
External calls:
- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
- IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
- IERC20(hermes).safeApprove(olympus,0) (Treasury.sol#1411)
- IERC20(hermes).safeApprove(olympus,_amount) (Treasury.sol#1412)
- IOlympus(olympus).allocateSeigniorage(_amount) (Treasury.sol#1413)
Event emitted after the call(s):
- OlympusFunded(now,_amount) (Treasury.sol#1414)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1427-1467):
External calls:
- _updateHermesPrice() (Treasury.sol#1428)
- IOracle(hermesOracle).update() (Treasury.sol#1320)
- _sendToOlympus(hermesSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1433)
- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#589)
- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
- IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
- IERC20(hermes).safeApprove(olympus,0) (Treasury.sol#1411)
- IERC20(hermes).safeApprove(olympus,_amount) (Treasury.sol#1412)
- IOlympus(olympus).allocateSeigniorage(_amount) (Treasury.sol#1413)
External calls sending eth:
- _sendToOlympus(hermesSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1433)
- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
Event emitted after the call(s):
- DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1392)
- _sendToOlympus(hermesSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1433)
- DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1399)
- _sendToOlympus(hermesSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1433)
```

```
	- OlympusFunded(now,_amount) (Treasury.sol#1414)
		- _sendToOlympus(hermesSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1433)
	- TeamFundFunded(now,_team1FundSharedAmount) (Treasury.sol#1406)
		- _sendToOlympus(hermesSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1433)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1427-1467):
	External calls:
	- _updateHermesPrice() (Treasury.sol#1428)
		- IOracle(hermesOracle).update() (Treasury.sol#1320)
	- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#589)
		- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
		- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
		- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
		- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
		- IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
		- IERC20(hermes).safeApprove(olympus,0) (Treasury.sol#1411)
		- IERC20(hermes).safeApprove(olympus,_amount) (Treasury.sol#1412)
		- IOlympus(olympus).allocateSeigniorage(_amount) (Treasury.sol#1413)
	External calls sending eth:
	- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
		- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
	Event emitted after the call(s):
	- DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1392)
		- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
	- DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1399)
		- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
	- OlympusFunded(now,_amount) (Treasury.sol#1414)
		- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
	- TeamFundFunded(now,_team1FundSharedAmount) (Treasury.sol#1406)
		- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1427-1467):
	External calls:
	- _updateHermesPrice() (Treasury.sol#1428)
		- IOracle(hermesOracle).update() (Treasury.sol#1320)
	- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#589)
		- IBasisAsset(hermes).mint(address(this),_amount) (Treasury.sol#1386)
```

```
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
⊠⊠- IERC20(hermes).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1391)
⊠⊠- IERC20(hermes).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1398)
⊠⊠- IERC20(hermes).transfer(team1Fund,_team1FundSharedAmount) (Treasury.sol#1405)
⊠⊠- IERC20(hermes).safeApprove(olympus,0) (Treasury.sol#1411)
⊠⊠- IERC20(hermes).safeApprove(olympus,_amount) (Treasury.sol#1412)
⊠⊠- IOlympus(olympus).allocateSeigniorage(_amount) (Treasury.sol#1413)
⊠- IBasisAsset(hermes).mint(address(this),_savedForBond) (Treasury.sol#1462)
⊠External calls sending eth:
⊠- _sendToOlympus(_savedForOlympus) (Treasury.sol#1458)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
⊠Event emitted after the call(s):
⊠- TreasuryFunded(now,_savedForBond) (Treasury.sol#1463)
Reentrancy in Treasury.buyBonds(uint256,uint256) (Treasury.sol#1330-1357):
⊠External calls:
⊠- IBasisAsset(hermes).burnFrom(msg.sender,_hermesAmount) (Treasury.sol#1350)
⊠- IBasisAsset(bbond).mint(msg.sender,_bondAmount) (Treasury.sol#1351)
⊠- _updateHermesPrice() (Treasury.sol#1354)
⊠⊠- IOracle(hermesOracle).update() (Treasury.sol#1320)
⊠Event emitted after the call(s):
⊠- BoughtBonds(msg.sender,_hermesAmount,_bondAmount) (Treasury.sol#1356)
Reentrancy in Treasury.redeemBonds(uint256,uint256) (Treasury.sol#1359-1383):
⊠External calls:
⊠- IBasisAsset(bbond).burnFrom(msg.sender,_bondAmount) (Treasury.sol#1377)
⊠- IERC20(hermes).safeTransfer(msg.sender,_hermesAmount) (Treasury.sol#1378)
⊠- _updateHermesPrice() (Treasury.sol#1380)
⊠⊠- IOracle(hermesOracle).update() (Treasury.sol#1320)
⊠Event emitted after the call(s):
⊠- RedeemedBonds(msg.sender,_hermesAmount,_bondAmount) (Treasury.sol#1382)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3

Address.isContract(address) (Treasury.sol#357-366) uses assembly
⊠- INLINE ASM (Treasury.sol#364)
Address._verifyCallResult(bool,bytes,string) (Treasury.sol#502-519) uses assembly
⊠- INLINE ASM (Treasury.sol#511-514)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
⊠- Version used: ['0.6.12', '>=0.6.0<0.8.0', '>=0.6.2<0.8.0', '^0.6.0']
⊠- >=0.6.0<0.8.0 (Treasury.sol#6)
```

```
- >=0.6.0<0.8.0 (Treasury.sol#39)
- >=0.6.0<0.8.0 (Treasury.sol#118)
- >=0.6.2<0.8.0 (Treasury.sol#334)
- >=0.6.0<0.8.0 (Treasury.sol#525)
- >=0.6.0<0.8.0 (Treasury.sol#600)
- ^0.6.0 (Treasury.sol#664)
- >=0.6.0<0.8.0 (Treasury.sol#685)
- >=0.6.0<0.8.0 (Treasury.sol#711)
- >=0.6.0<0.8.0 (Treasury.sol#716)
- 0.6.12 (Treasury.sol#785)
- 0.6.12 (Treasury.sol#825)
- ^0.6.0 (Treasury.sol#852)
- 0.6.12 (Treasury.sol#871)
- 0.6.12 (Treasury.sol#884)
- 0.6.12 (Treasury.sol#925)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used

Treasury._calculateMaxSupplyExpansionPercent(uint256) (Treasury.sol#1417-1425) has
costly operations inside a loop:
- maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1420)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-
operations-inside-a-loop

Address.functionCall(address,bytes) (Treasury.sol#410-412) is never used and should be
removed
Address.functionCallWithValue(address,bytes,uint256) (Treasury.sol#435-437) is never
used and should be removed
Address.functionDelegateCall(address,bytes) (Treasury.sol#484-486) is never used and
should be removed
Address.functionDelegateCall(address,bytes,string) (Treasury.sol#494-500) is never used
and should be removed
Address.functionStaticCall(address,bytes) (Treasury.sol#460-462) is never used and
should be removed
Address.functionStaticCall(address,bytes,string) (Treasury.sol#470-476) is never used
and should be removed
Address.sendValue(address,uint256) (Treasury.sol#384-390) is never used and should be
removed
Babylonian.sqrt(uint256) (Treasury.sol#667-679) is never used and should be removed
Context._msgData() (Treasury.sol#702-705) is never used and should be removed
Math.average(uint256,uint256) (Treasury.sol#30-33) is never used and should be removed
Math.max(uint256,uint256) (Treasury.sol#15-17) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (Treasury.sol#573-576) is never
used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (Treasury.sol#568-571) is never
used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (Treasury.sol#546-548) is
never used and should be removed
SafeMath.div(uint256,uint256,string) (Treasury.sol#305-308) is never used and should be
removed
SafeMath.mod(uint256,uint256) (Treasury.sol#267-270) is never used and should be
removed
SafeMath.mod(uint256,uint256,string) (Treasury.sol#325-328) is never used and should be
removed
SafeMath.sub(uint256,uint256,string) (Treasury.sol#285-288) is never used and should be
removed
SafeMath.tryAdd(uint256,uint256) (Treasury.sol#139-143) is never used and should be
removed
SafeMath.tryDiv(uint256,uint256) (Treasury.sol#175-178) is never used and should be
removed
SafeMath.tryMod(uint256,uint256) (Treasury.sol#185-188) is never used and should be
removed
SafeMath.tryMul(uint256,uint256) (Treasury.sol#160-168) is never used and should be
removed
SafeMath.trySub(uint256,uint256) (Treasury.sol#150-153) is never used and should be
removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.6.0<0.8.0 (Treasury.sol#6) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#39) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#118) is too complex
Pragma version>=0.6.2<0.8.0 (Treasury.sol#334) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#525) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#600) is too complex
Pragma version^0.6.0 (Treasury.sol#664) allows old versions
Pragma version>=0.6.0<0.8.0 (Treasury.sol#685) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#711) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#716) is too complex
Pragma version^0.6.0 (Treasury.sol#852) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
```

```
Low level call in Address.sendValue(address,uint256) (Treasury.sol#384-390):
	- (success) = recipient.call{value: amount}() (Treasury.sol#388)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string)
(Treasury.sol#445-452):
	- (success,returndata) = target.call{value: value}(data) (Treasury.sol#450)
Low level call in Address.functionStaticCall(address,bytes,string)
(Treasury.sol#470-476):
	- (success,returndata) = target.staticcall(data) (Treasury.sol#474)
Low level call in Address.functionDelegateCall(address,bytes,string)
(Treasury.sol#494-500):
	- (success,returndata) = target.delegatecall(data) (Treasury.sol#498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-
calls

Parameter Treasury.initialize(address,address,address,address,address,uint256)._hermes
(Treasury.sol#1158) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._bbond
(Treasury.sol#1159) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._bshare
(Treasury.sol#1160) is not in mixedCase
Parameter
Treasury.initialize(address,address,address,address,address,uint256)._hermesOracle
(Treasury.sol#1161) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._olympus
(Treasury.sol#1162) is not in mixedCase
Parameter
Treasury.initialize(address,address,address,address,address,uint256)._startTime
(Treasury.sol#1163) is not in mixedCase
Parameter Treasury.setOperator(address)._operator (Treasury.sol#1201) is not in
mixedCase
Parameter Treasury.setOlympus(address)._olympus (Treasury.sol#1205) is not in mixedCase
Parameter Treasury.setHermesOracle(address)._hermesOracle (Treasury.sol#1209) is not in
mixedCase
Parameter Treasury.setHermesPriceCeiling(uint256)._hermesPriceCeiling
(Treasury.sol#1213) is not in mixedCase
Parameter Treasury.setMaxSupplyExpansionPercents(uint256)._maxSupplyExpansionPercent
(Treasury.sol#1218) is not in mixedCase
Parameter Treasury.setSupplyTiersEntry(uint8,uint256)._index (Treasury.sol#1223) is not
in mixedCase
Parameter Treasury.setSupplyTiersEntry(uint8,uint256)._value (Treasury.sol#1223) is not
in mixedCase
```

```
Parameter Treasury.setMaxExpansionTiersEntry(uint8,uint256)._index (Treasury.sol#1236)
is not in mixedCase
Parameter Treasury.setMaxExpansionTiersEntry(uint8,uint256)._value (Treasury.sol#1236)
is not in mixedCase
Parameter Treasury.setBondDepletionFloorPercent(uint256)._bondDepletionFloorPercent
(Treasury.sol#1244) is not in mixedCase
Parameter Treasury.setMaxSupplyContractionPercent(uint256)._maxSupplyContractionPercent
(Treasury.sol#1249) is not in mixedCase
Parameter Treasury.setMaxDebtRatioPercent(uint256)._maxDebtRatioPercent
(Treasury.sol#1254) is not in mixedCase
Parameter Treasury.setBootstrap(uint256,uint256)._bootstrapEpochs (Treasury.sol#1259)
is not in mixedCase
Parameter Treasury.setBootstrap(uint256,uint256)._bootstrapSupplyExpansionPercent
(Treasury.sol#1259) is not in mixedCase
Parameter
Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._daoFund
(Treasury.sol#1267) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._daoFu
ndSharedPercent (Treasury.sol#1268) is not in mixedCase
Parameter
Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._devFund
(Treasury.sol#1269) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._devFu
ndSharedPercent (Treasury.sol#1270) is not in mixedCase
Parameter
Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._team1Fund
(Treasury.sol#1271) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._team1
FundSharedPercent (Treasury.sol#1272) is not in mixedCase
Parameter Treasury.setMaxDiscountRate(uint256)._maxDiscountRate (Treasury.sol#1288) is
not in mixedCase
Parameter Treasury.setMaxPremiumRate(uint256)._maxPremiumRate (Treasury.sol#1292) is
not in mixedCase
Parameter Treasury.setDiscountPercent(uint256)._discountPercent (Treasury.sol#1296) is
not in mixedCase
Parameter Treasury.setPremiumThreshold(uint256)._premiumThreshold (Treasury.sol#1301)
is not in mixedCase
Parameter Treasury.setPremiumPercent(uint256)._premiumPercent (Treasury.sol#1307) is
not in mixedCase
Parameter Treasury.setMintingFactorForPayingDebt(uint256)._mintingFactorForPayingDebt
(Treasury.sol#1312) is not in mixedCase
```

```
Parameter Treasury.buyBonds(uint256,uint256)._hermesAmount (Treasury.sol#1330) is not
in mixedCase
Parameter Treasury.redeemBonds(uint256,uint256)._bondAmount (Treasury.sol#1359) is not
in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._token
(Treasury.sol#1470) is not in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(Treasury.sol#1471) is not in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._to
(Treasury.sol#1472) is not in mixedCase
Parameter Treasury.olympusSetOperator(address)._operator (Treasury.sol#1481) is not in
mixedCase
Parameter Treasury.olympusSetLockUp(uint256,uint256)._withdrawLockupEpochs
(Treasury.sol#1485) is not in mixedCase
Parameter Treasury.olympusSetLockUp(uint256,uint256)._rewardLockupEpochs
(Treasury.sol#1485) is not in mixedCase
Parameter Treasury.olympusGovernanceRecoverUnsupported(address,uint256,address)._token
(Treasury.sol#1494) is not in mixedCase
Parameter Treasury.olympusGovernanceRecoverUnsupported(address,uint256,address)._amount
(Treasury.sol#1495) is not in mixedCase
Parameter Treasury.olympusGovernanceRecoverUnsupported(address,uint256,address)._to
(Treasury.sol#1496) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (Treasury.sol#703)" inContext (Treasury.sol#697-706)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

Variable Treasury.setExtraFunds(address,uint256,address,uint256,address,uint256)._daoFun
dSharedPercent (Treasury.sol#1268) is too similar to Treasury.setExtraFunds(address,uint
256,address,uint256,address,uint256)._devFundSharedPercent (Treasury.sol#1270)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
are-too-similar

Treasury.initialize(address,address,address,address,address,uint256)
(Treasury.sol#1157-1199) uses literals with too many digits:
- supplyTiers = (0,500000000000000000000,1000000000000000000000,150000000000000000000
00,2000000000000000000000,5000000000000000000000,10000000000000000000000,200000000000
00000000000,50000000000000000000000) (Treasury.sol#1176)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits
```

```
renounceOwnership() should be declared external:
⊠- Ownable.renounceOwnership() (Treasury.sol#766-769)
transferOwnership(address) should be declared external:
⊠- Ownable.transferOwnership(address) (Treasury.sol#775-779)
operator() should be declared external:
⊠- Operator.operator() (Treasury.sol#798-800)
isOperator() should be declared external:
⊠- Operator.isOperator() (Treasury.sol#807-809)
transferOperator(address) should be declared external:
⊠- Operator.transferOperator(address) (Treasury.sol#811-813)
isInitialized() should be declared external:
⊠- Treasury.isInitialized() (Treasury.sol#1064-1066)
getHermesUpdatedPrice() should be declared external:
⊠- Treasury.getHermesUpdatedPrice() (Treasury.sol#1082-1088)
getReserve() should be declared external:
⊠- Treasury.getReserve() (Treasury.sol#1091-1093)
getBurnableHermesLeft() should be declared external:
⊠- Treasury.getBurnableHermesLeft() (Treasury.sol#1095-1107)
getRedeemableBonds() should be declared external:
⊠- Treasury.getRedeemableBonds() (Treasury.sol#1109-1118)
initialize(address,address,address,address,address,uint256) should be declared
external:
⊠- Treasury.initialize(address,address,address,address,address,uint256)
(Treasury.sol#1157-1199)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
```

Guard