0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

Draco Finance

February 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

This report has been prepared for the Draco Finance team upon their request.

The audited project is a fork of the Tomb Finance Project.

The purpose of this 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 prior to deployment.

Further details about Draco Finance are available at the official website: https://www.draco.finance.

Name	Draco Finance
Audit date	2022-02-17 - 2022-02-17
Language	Solidity
Platform	Fantom Network

Contracts checked

Name	Address
SDraco	0x713A18d059EA1D12E5bE134a864C075E47d5FEFA
DBond	0x6d3e602b88d6Add9817930803BE766ED9179bF02
DracoGenesisRewardPool	0xB5cd7B1fD153c6FBf6F5219721a296Fc2b69f2F5
TaxOfficeV2	0x628534d380712FB9bA1eA33f91967E8049D9E035
Masonry	0x39AEd2eC961AA9da9D778C80B6f90CD80dBFAE16
Draco	0x37863ea4bf6ef836bC8bE909221BAF09A2aF43d7
SDracoRewardPool	0x628534d380712FB9bA1eA33f91967E8049D9E035
Treasury	0x76344B0cD69b9772297304070cE01C356065b379

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 Draco Finance Project was compared with the Tomb Project. Draco Finance has changed the implementation of Treasury and Token contracts.

The changed 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 0x37863ea4bf6ef836bC8bE909221BAF09A2aF43d7.Also, new state variables were declared: INITIAL_TOMB_POOL_DISTRIBUTION and INITIAL_AIRDROP_WALLET_DISTRIBUTION. Their values represent amounts of tokens being distributed during distributeReward function invoking.

In the contract Treasury the array of pools excludedFromTotalSupplywas removed.

Disclaimer

Static code analysis results

```
INFO:Detectors:
UniswapV2OracleLibrary.currentBlockTimestamp() (contracts/lib/
UniswapV2OracleLibrary.sol#13-15) uses a weak PRNG: "uint32(block.timestamp % 2 ** 32)
(contracts/lib/UniswapV2OracleLibrary.sol#14)"
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#weak-PRNG
INFO:Detectors:
SDracoRewardPool is re-used:
        - contracts/distribution/SDracoRewardPool.sol#11-274
        - contracts/SDracoRewardPool.sol#11-274
IERC20 is re-used:
        - contracts/interfaces/IERC20.sol#8-77
        - node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#8-77
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#name-reused
INFO:Detectors:
Draco.governanceRecoverUnsupported(IERC20,uint256,address) (contracts/
Draco.sol#265-271) ignores return value by _token.transfer(_to,_amount) (contracts/
Draco.sol#270)
SDraco.governanceRecoverUnsupported(IERC20,uint256,address) (contracts/
SDraco.sol#116-122) ignores return value by _token.transfer(_to,_amount) (contracts/
SDraco.sol#121)
TaxOfficeV2.addLiquidityTaxFree(address,uint256,uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#88-133) ignores return value by
IERC20(draco).transferFrom(msg.sender,address(this),amtDraco) (contracts/
TaxOfficeV2.sol#105)
TaxOfficeV2.addLiquidityTaxFree(address,uint256,uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#88-133) ignores return value by
IERC20(token).transferFrom(msg.sender,address(this),amtToken) (contracts/
TaxOfficeV2.sol#106)
TaxOfficeV2.addLiquidityTaxFree(address,uint256,uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#88-133) ignores return value by
IERC20(draco).transfer(msg.sender,amtDraco.sub(resultAmtDraco)) (contracts/
TaxOfficeV2.sol#127)
TaxOfficeV2.addLiquidityTaxFree(address,uint256,uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#88-133) ignores return value by
IERC20(token).transfer(msg.sender,amtToken.sub(resultAmtToken)) (contracts/
TaxOfficeV2.sol#130)
TaxOfficeV2.addLiquidityETHTaxFree(uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#135-172) ignores return value by
IERC20(draco).transferFrom(msg.sender,address(this),amtDraco) (contracts/
TaxOfficeV2.sol#151)
TaxOfficeV2.addLiquidityETHTaxFree(uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#135-172) ignores return value by
IERC20(draco).transfer(msg.sender,amtDraco.sub(resultAmtDraco)) (contracts/
TaxOfficeV2.sol#169)
TaxOfficeV2.taxFreeTransferFrom(address,address,uint256) (contracts/
TaxOfficeV2.sol#182-191) ignores return value by
IERC20(draco).transferFrom(_sender,_recipient,_amt) (contracts/TaxOfficeV2.sol#189)
Treasury._sendToMasonry(uint256) (contracts/Treasury.sol#457-480) ignores return value
by IERC20(draco).transfer(daoFund,_daoFundSharedAmount) (contracts/Treasury.sol#463)
Treasury._sendToMasonry(uint256) (contracts/Treasury.sol#457-480) ignores return value
by IERC20(draco).transfer(devFund,_devFundSharedAmount) (contracts/Treasury.sol#470)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer
INFO:Detectors:
SDracoRewardPool.pendingShare(uint256,address) (contracts/SDracoRewardPool.sol#150-161)
performs a multiplication on the result of a division:
        -_sdracoReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(contracts/SDracoRewardPool.sol#157)
        -accSDracoPerShare =
accSDracoPerShare.add(_sdracoReward.mul(1e18).div(tokenSupply)) (contracts/
SDracoRewardPool.sol#158)
SDracoRewardPool.updatePool(uint256) (contracts/SDracoRewardPool.sol#172-192) performs
a multiplication on the result of a division:
        -_sdracoReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(contracts/SDracoRewardPool.sol#188)
        -pool.accSDracoPerShare =
pool.accSDracoPerShare.add(_sdracoReward.mul(1e18).div(tokenSupply)) (contracts/
SDracoRewardPool.sol#189)
Treasury.allocateSeigniorage() (contracts/Treasury.sol#492-532) performs a
multiplication on the result of a division:
        -_seigniorage = dracoSupply.mul(_percentage).div(1e18) (contracts/
Treasury.sol#515)
        -_savedForMasonry =
_seigniorage.mul(seigniorageExpansionFloorPercent).div(10000) (contracts/
Treasury.sol#516)
DracoGenesisRewardPool.pendingDRACO(uint256,address) (contracts/distribution/
DracoGenesisRewardPool.sol#167-189) performs a multiplication on the result of a
division:
        -_dracoReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(contracts/distribution/DracoGenesisRewardPool.sol#181-183)
        -accDracoPerShare =
accDracoPerShare.add(_dracoReward.mul(1e18).div(tokenSupply)) (contracts/distribution/
DracoGenesisRewardPool.sol#184-186)
DracoGenesisRewardPool.updatePool(uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#200-227) performs a multiplication on the result of a
division:
        -_dracoReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(contracts/distribution/DracoGenesisRewardPool.sol#219-221)
        -pool.accDracoPerShare =
pool.accDracoPerShare.add(_dracoReward.mul(1e18).div(tokenSupply)) (contracts/
distribution/DracoGenesisRewardPool.sol#222-224)
DracoRewardPool.pendingDRACO(uint256,address) (contracts/distribution/
DracoRewardPool.sol#156-167) performs a multiplication on the result of a division:
        -_dracoReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(contracts/distribution/DracoRewardPool.sol#163)
        -accDracoPerShare =
accDracoPerShare.add(_dracoReward.mul(1e18).div(tokenSupply)) (contracts/distribution/
DracoRewardPool.sol#164)
DracoRewardPool.updatePool(uint256) (contracts/distribution/
DracoRewardPool.sol#178-198) performs a multiplication on the result of a division:
        -_dracoReward = _generatedReward.mul(pool.allocPoint).div(totalAllocPoint)
(contracts/distribution/DracoRewardPool.sol#194)
        -pool.accDracoPerShare =
pool.accDracoPerShare.add(_dracoReward.mul(1e18).div(tokenSupply)) (contracts/
distribution/DracoRewardPool.sol#195)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply
INFO:Detectors:
SDracoRewardPool.updatePool(uint256) (contracts/SDracoRewardPool.sol#172-192) uses a
dangerous strict equality:
        - tokenSupply == 0 (contracts/SDracoRewardPool.sol#178)
DracoGenesisRewardPool.updatePool(uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#200-227) uses a dangerous strict equality:
        - tokenSupply == 0 (contracts/distribution/DracoGenesisRewardPool.sol#206)
DracoRewardPool.updatePool(uint256) (contracts/distribution/
DracoRewardPool.sol#178-198) uses a dangerous strict equality:
        - tokenSupply == 0 (contracts/distribution/DracoRewardPool.sol#184)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-
strict-equalities
```

```
INFO:Detectors:
Reentrancy in Treasury.buyBonds(uint256,uint256) (contracts/Treasury.sol#402-429):
        External calls:
        - IBasisAsset(draco).burnFrom(msg.sender,_dracoAmount) (contracts/
Treasury.sol#422)
        - IBasisAsset(dbond).mint(msg.sender,_bondAmount) (contracts/Treasury.sol#423)
        State variables written after the call(s):
        - epochSupplyContractionLeft = epochSupplyContractionLeft.sub(_dracoAmount)
(contracts/Treasury.sol#425)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-1
INFO:Detectors:
Draco.setTaxTiersTwap(uint8,uint256) (contracts/Draco.sol#95-106) contains a tautology
or contradiction:
        - require(bool,string)(_index >= 0,Index has to be higher than 0) (contracts/
Draco.sol#96)
Draco.setTaxTiersRate(uint8,uint256) (contracts/Draco.sol#108-113) contains a tautology
or contradiction:
        - require(bool,string)(_index >= 0,Index has to be higher than 0) (contracts/
Draco.sol#109)
Draco._updateTaxRate(uint256) (contracts/Draco.sol#127-137) contains a tautology or
contradiction:
        - tierId >= 0 (contracts/Draco.sol#129)
Treasury.setSupplyTiersEntry(uint8,uint256) (contracts/Treasury.sol#302-313) contains a
tautology or contradiction:
        - require(bool,string)(_index >= 0,Index has to be higher than 0) (contracts/
Treasury.sol#303)
Treasury.setMaxExpansionTiersEntry(uint8,uint256) (contracts/Treasury.sol#315-321)
contains a tautology or contradiction:
        - require(bool,string)(_index >= 0,Index has to be higher than 0) (contracts/
Treasury.sol#316)
Treasury._calculateMaxSupplyExpansionPercent(uint256) (contracts/Treasury.sol#482-490)
contains a tautology or contradiction:
        - tierId >= 0 (contracts/Treasury.sol#483)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#tautology-or-
contradiction
INFO:Detectors:
FixedPoint.mul(FixedPoint.uq112x112,uint256).z (contracts/lib/FixedPoint.sol#44) is a
local variable never initialized
Treasury.allocateSeigniorage()._savedForBond (contracts/Treasury.sol#504) is a local
variable never initialized
```

```
UniswapV2Library.getAmountsOut(address,uint256,address[]).i (contracts/lib/
UniswapV2Library.sol#97) is a local variable never initialized
Draco._getDracoPrice()._price (contracts/Draco.sol#120) is a local variable never
initialized
Treasury.getDracoPrice().price (contracts/Treasury.sol#154) is a local variable never
initialized
Treasury.getDracoUpdatedPrice().price (contracts/Treasury.sol#162) is a local variable
never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-
local-variables
INFO:Detectors:
Draco._getDracoPrice() (contracts/Draco.sol#119-125) ignores return value by
IOracle(dracoOracle).consult(address(this),1e18) (contracts/Draco.sol#120-124)
TaxOfficeV2._approveTokenIfNeeded(address,address) (contracts/TaxOfficeV2.sol#197-201)
ignores return value by IERC20(_token).approve(_router,type()(uint256).max) (contracts/
TaxOfficeV2.sol#199)
Treasury.getDracoPrice() (contracts/Treasury.sol#153-159) ignores return value by
IOracle(dracoOracle).consult(draco,1e18) (contracts/Treasury.sol#154-158)
Treasury.getDracoUpdatedPrice() (contracts/Treasury.sol#161-167) ignores return value
by IOracle(dracoOracle).twap(draco,1e18) (contracts/Treasury.sol#162-166)
Treasury.buyBonds(uint256,uint256) (contracts/Treasury.sol#402-429) ignores return
value by IBasisAsset(dbond).mint(msg.sender,_bondAmount) (contracts/Treasury.sol#423)
Treasury._sendToMasonry(uint256) (contracts/Treasury.sol#457-480) ignores return value
by IBasisAsset(draco).mint(address(this),_amount) (contracts/Treasury.sol#458)
Treasury.allocateSeigniorage() (contracts/Treasury.sol#492-532) ignores return value by
IBasisAsset(draco).mint(address(this),_savedForBond) (contracts/Treasury.sol#527)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
Masonry.setOperator(address) (contracts/Masonry.sol#146-148) should emit an event for:
        - operator = _operator (contracts/Masonry.sol#147)
SDracoRewardPool.setOperator(address) (contracts/SDracoRewardPool.sol#258-260) should
emit an event for:
        - operator = _operator (contracts/SDracoRewardPool.sol#259)
Treasury.setOperator(address) (contracts/Treasury.sol#280-282) should emit an event
for:
        - operator = _operator (contracts/Treasury.sol#281)
Treasury.setMasonry(address) (contracts/Treasury.sol#284-286) should emit an event
for:
        - masonry = _masonry (contracts/Treasury.sol#285)
DracoGenesisRewardPool.setOperator(address) (contracts/distribution/
DracoGenesisRewardPool.sol#305-307) should emit an event for:
        - operator = _operator (contracts/distribution/DracoGenesisRewardPool.sol#306)
DracoRewardPool.setOperator(address) (contracts/distribution/
DracoRewardPool.sol#264-266) should emit an event for:
        - operator = _operator (contracts/distribution/DracoRewardPool.sol#265)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
access-control
INFO:Detectors:
Draco.setBurnThreshold(uint256) (contracts/Draco.sol#115-117) should emit an event
for:
        - burnThreshold = _burnThreshold (contracts/Draco.sol#116)
Draco.setTaxRate(uint256) (contracts/Draco.sol#163-167) should emit an event for:
        - taxRate = _taxRate (contracts/Draco.sol#166)
Masonry.setLockUp(uint256,uint256) (contracts/Masonry.sol#150-154) should emit an event
for:
        - withdrawLockupEpochs = _withdrawLockupEpochs (contracts/Masonry.sol#152)
        - rewardLockupEpochs = _rewardLockupEpochs (contracts/Masonry.sol#153)
SDracoRewardPool.add(uint256,IERC20,bool,uint256) (contracts/
SDracoRewardPool.sol#83-121) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (contracts/
SDracoRewardPool.sol#119)
SDracoRewardPool.set(uint256,uint256) (contracts/SDracoRewardPool.sol#124-133) should
emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint)
(contracts/SDracoRewardPool.sol#128-130)
Treasury.setDracoPriceCeiling(uint256) (contracts/Treasury.sol#292-295) should emit an
event for:
        - dracoPriceCeiling = _dracoPriceCeiling (contracts/Treasury.sol#294)
Treasury.setMaxSupplyExpansionPercents(uint256) (contracts/Treasury.sol#297-300) should
emit an event for:
        - maxSupplyExpansionPercent = _maxSupplyExpansionPercent (contracts/
Treasury.sol#299)
Treasury.setBondDepletionFloorPercent(uint256) (contracts/Treasury.sol#323-326) should
emit an event for:
        - bondDepletionFloorPercent = _bondDepletionFloorPercent (contracts/
Treasury.sol#325)
Treasury.setMaxDebtRatioPercent(uint256) (contracts/Treasury.sol#333-336) should emit
an event for:
        - maxDebtRatioPercent = _maxDebtRatioPercent (contracts/Treasury.sol#335)
Treasury.setBootstrap(uint256,uint256) (contracts/Treasury.sol#338-343) should emit an
event for:
        - bootstrapEpochs = _bootstrapEpochs (contracts/Treasury.sol#341)
        - bootstrapSupplyExpansionPercent = _bootstrapSupplyExpansionPercent (contracts/
Treasury.sol#342)
Treasury.setExtraFunds(address,uint256,address,uint256) (contracts/
Treasury.sol#345-359) should emit an event for:
        - daoFundSharedPercent = _daoFundSharedPercent (contracts/Treasury.sol#356)
        - devFundSharedPercent = _devFundSharedPercent (contracts/Treasury.sol#358)
Treasury.setMaxDiscountRate(uint256) (contracts/Treasury.sol#361-363) should emit an
event for:
        - maxDiscountRate = _maxDiscountRate (contracts/Treasury.sol#362)
Treasury.setMaxPremiumRate(uint256) (contracts/Treasury.sol#365-367) should emit an
event for:
        - maxPremiumRate = _maxPremiumRate (contracts/Treasury.sol#366)
Treasury.setDiscountPercent(uint256) (contracts/Treasury.sol#369-372) should emit an
event for:
        - discountPercent = _discountPercent (contracts/Treasury.sol#371)
Treasury.setPremiumThreshold(uint256) (contracts/Treasury.sol#374-378) should emit an
event for:
        - premiumThreshold = _premiumThreshold (contracts/Treasury.sol#377)
Treasury.setPremiumPercent(uint256) (contracts/Treasury.sol#380-383) should emit an
event for:
        - premiumPercent = _premiumPercent (contracts/Treasury.sol#382)
Treasury.setMintingFactorForPayingDebt(uint256) (contracts/Treasury.sol#385-388) should
emit an event for:
        - mintingFactorForPayingDebt = _mintingFactorForPayingDebt (contracts/
Treasury.sol#387)
DracoGenesisRewardPool.add(uint256,IERC20,bool,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#93-132) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (contracts/distribution/
DracoGenesisRewardPool.sol#130)
DracoGenesisRewardPool.set(uint256,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#135-144) should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint)
(contracts/distribution/DracoGenesisRewardPool.sol#139-141)
DracoRewardPool.add(uint256,IERC20,bool,uint256) (contracts/distribution/
DracoRewardPool.sol#89-119) should emit an event for:
        - totalAllocPoint = totalAllocPoint.add(_allocPoint) (contracts/distribution/
DracoRewardPool.sol#117)
DracoRewardPool.set(uint256,uint256) (contracts/distribution/
DracoRewardPool.sol#122-129) should emit an event for:
        - totalAllocPoint = totalAllocPoint.sub(pool.allocPoint).add(_allocPoint)
(contracts/distribution/DracoRewardPool.sol#126)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic
INFO:Detectors:
Masonry.setOperator(address)._operator (contracts/Masonry.sol#146) lacks a zero-check
on :
                - operator = _operator (contracts/Masonry.sol#147)
SDraco.setTreasuryFund(address)._communityFund (contracts/SDraco.sol#61) lacks a zero-
check on :
                - communityFund = _communityFund (contracts/SDraco.sol#63)
SDracoRewardPool.setOperator(address)._operator (contracts/SDracoRewardPool.sol#258)
lacks a zero-check on :
                - operator = _operator (contracts/SDracoRewardPool.sol#259)
Treasury.initialize(address,address,address,address,address,uint256)._draco (contracts/
Treasury.sol#237) lacks a zero-check on :
                - draco = _draco (contracts/Treasury.sol#244)
Treasury.initialize(address,address,address,address,address,uint256)._dbond (contracts/
Treasury.sol#238) lacks a zero-check on :
                - dbond = _dbond (contracts/Treasury.sol#245)
Treasury.initialize(address,address,address,address,address,uint256)._sdraco (contracts/
Treasury.sol#239) lacks a zero-check on :
                - sdraco = _sdraco (contracts/Treasury.sol#246)
Treasury.initialize(address,address,address,address,address,uint256)._dracoOracle
(contracts/Treasury.sol#240) lacks a zero-check on :
                - dracoOracle = _dracoOracle (contracts/Treasury.sol#247)
Treasury.initialize(address,address,address,address,address,uint256)._masonry
(contracts/Treasury.sol#241) lacks a zero-check on :
                - masonry = _masonry (contracts/Treasury.sol#248)
Treasury.setOperator(address)._operator (contracts/Treasury.sol#280) lacks a zero-check
on :
                - operator = _operator (contracts/Treasury.sol#281)
Treasury.setMasonry(address)._masonry (contracts/Treasury.sol#284) lacks a zero-check
on :
                - masonry = _masonry (contracts/Treasury.sol#285)
Treasury.setDracoOracle(address)._dracoOracle (contracts/Treasury.sol#288) lacks a zero-
check on :
                - dracoOracle = _dracoOracle (contracts/Treasury.sol#289)
DracoGenesisRewardPool.setOperator(address)._operator (contracts/distribution/
DracoGenesisRewardPool.sol#305) lacks a zero-check on :
                - operator = _operator (contracts/distribution/
DracoGenesisRewardPool.sol#306)
DracoRewardPool.setOperator(address)._operator (contracts/distribution/
DracoRewardPool.sol#264) lacks a zero-check on :
                - operator = _operator (contracts/distribution/DracoRewardPool.sol#265)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation
INFO:Detectors:
Distributor.distribute() (contracts/Distributor.sol#14-18) has external calls inside a
loop: distributors[i].distribute() (contracts/Distributor.sol#16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-
a-loop
INFO:Detectors:
Variable 'Draco._getDracoPrice()._price (contracts/Draco.sol#120)' in
Draco._getDracoPrice() (contracts/Draco.sol#119-125) potentially used before
declaration: uint256(_price) (contracts/Draco.sol#121)
Variable 'Treasury.getDracoPrice().price (contracts/Treasury.sol#154)' in
Treasury.getDracoPrice() (contracts/Treasury.sol#153-159) potentially used before
declaration: uint256(price) (contracts/Treasury.sol#155)
Variable 'Treasury.getDracoUpdatedPrice().price (contracts/Treasury.sol#162)' in
Treasury.getDracoUpdatedPrice() (contracts/Treasury.sol#161-167) potentially used
before declaration: uint256(price) (contracts/Treasury.sol#163)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-
declaration-usage-of-local-variables
INFO:Detectors:
Reentrancy in Treasury.allocateSeigniorage() (contracts/Treasury.sol#492-532):
        External calls:
        - _updateDracoPrice() (contracts/Treasury.sol#493)
                - IOracle(dracoOracle).update() (contracts/Treasury.sol#393)
        State variables written after the call(s):
        - _mse = _calculateMaxSupplyExpansionPercent(dracoSupply).mul(1e14) (contracts/
Treasury.sol#506)
                - maxSupplyExpansionPercent = maxExpansionTiers[tierId] (contracts/
Treasury.sol#485)
        - previousEpochDracoPrice = getDracoPrice() (contracts/Treasury.sol#494)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-2
INFO:Detectors:
Reentrancy in Treasury._sendToMasonry(uint256) (contracts/Treasury.sol#457-480):
        External calls:
        - IBasisAsset(draco).mint(address(this),_amount) (contracts/Treasury.sol#458)
        - IERC20(draco).transfer(daoFund,_daoFundSharedAmount) (contracts/
Treasury.sol#463)
        Event emitted after the call(s):
```

```
        - DaoFundFunded(now,_daoFundSharedAmount) (contracts/Treasury.sol#464)
Reentrancy in Treasury._sendToMasonry(uint256) (contracts/Treasury.sol#457-480):
        External calls:
        - IBasisAsset(draco).mint(address(this),_amount) (contracts/Treasury.sol#458)
        - IERC20(draco).transfer(daoFund,_daoFundSharedAmount) (contracts/
Treasury.sol#463)
        - IERC20(draco).transfer(devFund,_devFundSharedAmount) (contracts/
Treasury.sol#470)
        Event emitted after the call(s):
        - DevFundFunded(now,_devFundSharedAmount) (contracts/Treasury.sol#471)
Reentrancy in Treasury._sendToMasonry(uint256) (contracts/Treasury.sol#457-480):
        External calls:
        - IBasisAsset(draco).mint(address(this),_amount) (contracts/Treasury.sol#458)
        - IERC20(draco).transfer(daoFund,_daoFundSharedAmount) (contracts/
Treasury.sol#463)
        - IERC20(draco).transfer(devFund,_devFundSharedAmount) (contracts/
Treasury.sol#470)
        - IERC20(draco).safeApprove(masonry,0) (contracts/Treasury.sol#476)
        - IERC20(draco).safeApprove(masonry,_amount) (contracts/Treasury.sol#477)
        - IMasonry(masonry).allocateSeigniorage(_amount) (contracts/Treasury.sol#478)
        Event emitted after the call(s):
        - MasonryFunded(now,_amount) (contracts/Treasury.sol#479)
Reentrancy in Masonry.allocateSeigniorage(uint256) (contracts/Masonry.sol#241-258):
        External calls:
        - draco.safeTransferFrom(msg.sender,address(this),amount) (contracts/
Masonry.sol#256)
        Event emitted after the call(s):
        - RewardAdded(msg.sender,amount) (contracts/Masonry.sol#257)
Reentrancy in Treasury.buyBonds(uint256,uint256) (contracts/Treasury.sol#402-429):
        External calls:
        - IBasisAsset(draco).burnFrom(msg.sender,_dracoAmount) (contracts/
Treasury.sol#422)
        - IBasisAsset(dbond).mint(msg.sender,_bondAmount) (contracts/Treasury.sol#423)
        - _updateDracoPrice() (contracts/Treasury.sol#426)
                - IOracle(dracoOracle).update() (contracts/Treasury.sol#393)
        Event emitted after the call(s):
        - BoughtBonds(msg.sender,_dracoAmount,_bondAmount) (contracts/Treasury.sol#428)
Reentrancy in Masonry.claimReward() (contracts/Masonry.sol#230-239):
        External calls:
        - draco.safeTransfer(msg.sender,reward) (contracts/Masonry.sol#236)
        Event emitted after the call(s):
```

```
        - RewardPaid(msg.sender,reward) (contracts/Masonry.sol#237)
Reentrancy in SDracoRewardPool.emergencyWithdraw(uint256) (contracts/
SDracoRewardPool.sol#236-244):
        External calls:
        - pool.token.safeTransfer(msg.sender,_amount) (contracts/
SDracoRewardPool.sol#242)
        Event emitted after the call(s):
        - EmergencyWithdraw(msg.sender,_pid,_amount) (contracts/
SDracoRewardPool.sol#243)
Reentrancy in DracoGenesisRewardPool.emergencyWithdraw(uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#283-291):
        External calls:
        - pool.token.safeTransfer(msg.sender,_amount) (contracts/distribution/
DracoGenesisRewardPool.sol#289)
        Event emitted after the call(s):
        - EmergencyWithdraw(msg.sender,_pid,_amount) (contracts/distribution/
DracoGenesisRewardPool.sol#290)
Reentrancy in DracoRewardPool.emergencyWithdraw(uint256) (contracts/distribution/
DracoRewardPool.sol#242-250):
        External calls:
        - pool.token.safeTransfer(msg.sender,_amount) (contracts/distribution/
DracoRewardPool.sol#248)
        Event emitted after the call(s):
        - EmergencyWithdraw(msg.sender,_pid,_amount) (contracts/distribution/
DracoRewardPool.sol#249)
Reentrancy in Treasury.redeemBonds(uint256,uint256) (contracts/Treasury.sol#431-455):
        External calls:
        - IBasisAsset(dbond).burnFrom(msg.sender,_bondAmount) (contracts/
Treasury.sol#449)
        - IERC20(draco).safeTransfer(msg.sender,_dracoAmount) (contracts/
Treasury.sol#450)
        - _updateDracoPrice() (contracts/Treasury.sol#452)
                - IOracle(dracoOracle).update() (contracts/Treasury.sol#393)
        Event emitted after the call(s):
        - RedeemedBonds(msg.sender,_dracoAmount,_bondAmount) (contracts/
Treasury.sol#454)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3
INFO:Detectors:
SDraco.unclaimedTreasuryFund() (contracts/SDraco.sol#72-77) uses timestamp for
comparisons
```

```
        Dangerous comparisons:
        - _now > endTime (contracts/SDraco.sol#74)
        - communityFundLastClaimed >= _now (contracts/SDraco.sol#75)
SDraco.unclaimedDevFund() (contracts/SDraco.sol#79-84) uses timestamp for comparisons
        Dangerous comparisons:
        - _now > endTime (contracts/SDraco.sol#81)
        - devFundLastClaimed >= _now (contracts/SDraco.sol#82)
SDracoRewardPool.constructor(address,uint256) (contracts/SDracoRewardPool.sol#59-68)
uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp < _poolStartTime,late) (contracts/
SDracoRewardPool.sol#63)
SDracoRewardPool.checkPoolDuplicate(IERC20) (contracts/SDracoRewardPool.sol#75-80) uses
timestamp for comparisons
        Dangerous comparisons:
        - pid < length (contracts/SDracoRewardPool.sol#77)
        - require(bool,string)(poolInfo[pid].token != _token,SDracoRewardPool: existing
pool?) (contracts/SDracoRewardPool.sol#78)
SDracoRewardPool.add(uint256,IERC20,bool,uint256) (contracts/
SDracoRewardPool.sol#83-121) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolStartTime (contracts/SDracoRewardPool.sol#93)
        - _lastRewardTime == 0 (contracts/SDracoRewardPool.sol#95)
        - _lastRewardTime < poolStartTime (contracts/SDracoRewardPool.sol#98)
        - _lastRewardTime == 0 || _lastRewardTime < block.timestamp (contracts/
SDracoRewardPool.sol#104)
        - _isStarted = (_lastRewardTime <= poolStartTime) || (_lastRewardTime <=
block.timestamp) (contracts/SDracoRewardPool.sol#108-110)
SDracoRewardPool.getGeneratedReward(uint256,uint256) (contracts/
SDracoRewardPool.sol#136-147) uses timestamp for comparisons
        Dangerous comparisons:
        - _fromTime >= _toTime (contracts/SDracoRewardPool.sol#137)
        - _toTime >= poolEndTime (contracts/SDracoRewardPool.sol#138)
        - _toTime <= poolStartTime (contracts/SDracoRewardPool.sol#143)
SDracoRewardPool.pendingShare(uint256,address) (contracts/SDracoRewardPool.sol#150-161)
uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > pool.lastRewardTime && tokenSupply != 0 (contracts/
SDracoRewardPool.sol#155)
SDracoRewardPool.massUpdatePools() (contracts/SDracoRewardPool.sol#164-169) uses
timestamp for comparisons
```

```
        Dangerous comparisons:
        - pid < length (contracts/SDracoRewardPool.sol#166)
SDracoRewardPool.updatePool(uint256) (contracts/SDracoRewardPool.sol#172-192) uses
timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= pool.lastRewardTime (contracts/SDracoRewardPool.sol#174)
SDracoRewardPool.governanceRecoverUnsupported(IERC20,uint256,address) (contracts/
SDracoRewardPool.sol#262-273) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolEndTime + 7776000 (contracts/SDracoRewardPool.sol#263)
TaxOfficeV2.addLiquidityTaxFree(address,uint256,uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#88-133) uses timestamp for comparisons
        Dangerous comparisons:
        - amtDraco.sub(resultAmtDraco) > 0 (contracts/TaxOfficeV2.sol#126)
        - amtToken.sub(resultAmtToken) > 0 (contracts/TaxOfficeV2.sol#129)
TaxOfficeV2.addLiquidityETHTaxFree(uint256,uint256,uint256) (contracts/
TaxOfficeV2.sol#135-172) uses timestamp for comparisons
        Dangerous comparisons:
        - amtDraco.sub(resultAmtDraco) > 0 (contracts/TaxOfficeV2.sol#168)
DracoGenesisRewardPool.constructor(address,address,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#65-72) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp < _poolStartTime,late) (contracts/
distribution/DracoGenesisRewardPool.sol#66)
DracoGenesisRewardPool.checkPoolDuplicate(IERC20) (contracts/distribution/
DracoGenesisRewardPool.sol#82-90) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (contracts/distribution/DracoGenesisRewardPool.sol#84)
        - require(bool,string)(poolInfo[pid].token != _token,DracoGenesisPool: existing
pool?) (contracts/distribution/DracoGenesisRewardPool.sol#85-88)
DracoGenesisRewardPool.add(uint256,IERC20,bool,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#93-132) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolStartTime (contracts/distribution/
DracoGenesisRewardPool.sol#103)
        - _lastRewardTime == 0 (contracts/distribution/DracoGenesisRewardPool.sol#105)
        - _lastRewardTime < poolStartTime (contracts/distribution/
DracoGenesisRewardPool.sol#108)
        - _lastRewardTime == 0 || _lastRewardTime < block.timestamp (contracts/
distribution/DracoGenesisRewardPool.sol#114)
        - _isStarted = (_lastRewardTime <= poolStartTime) || (_lastRewardTime <=
block.timestamp) (contracts/distribution/DracoGenesisRewardPool.sol#118-119)
DracoGenesisRewardPool.getGeneratedReward(uint256,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#147-164) uses timestamp for comparisons
        Dangerous comparisons:
        - _fromTime >= _toTime (contracts/distribution/DracoGenesisRewardPool.sol#152)
        - _toTime >= poolEndTime (contracts/distribution/
DracoGenesisRewardPool.sol#153)
        - _toTime <= poolStartTime (contracts/distribution/
DracoGenesisRewardPool.sol#159)
DracoGenesisRewardPool.pendingDRACO(uint256,address) (contracts/distribution/
DracoGenesisRewardPool.sol#167-189) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > pool.lastRewardTime && tokenSupply != 0 (contracts/
distribution/DracoGenesisRewardPool.sol#176)
DracoGenesisRewardPool.massUpdatePools() (contracts/distribution/
DracoGenesisRewardPool.sol#192-197) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (contracts/distribution/DracoGenesisRewardPool.sol#194)
DracoGenesisRewardPool.updatePool(uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#200-227) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= pool.lastRewardTime (contracts/distribution/
DracoGenesisRewardPool.sol#202)
DracoGenesisRewardPool.governanceRecoverUnsupported(IERC20,uint256,address) (contracts/
distribution/DracoGenesisRewardPool.sol#309-324) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < poolEndTime + 2592000 (contracts/distribution/
DracoGenesisRewardPool.sol#314)
DracoRewardPool.constructor(address,uint256) (contracts/distribution/
DracoRewardPool.sol#60-74) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp < _poolStartTime,late) (contracts/
distribution/DracoRewardPool.sol#61)
DracoRewardPool.checkPoolDuplicate(IERC20) (contracts/distribution/
DracoRewardPool.sol#81-86) uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (contracts/distribution/DracoRewardPool.sol#83)
        - require(bool,string)(poolInfo[pid].token != _token,DracoRewardPool: existing
pool?) (contracts/distribution/DracoRewardPool.sol#84)
DracoRewardPool.add(uint256,IERC20,bool,uint256) (contracts/distribution/
DracoRewardPool.sol#89-119) uses timestamp for comparisons
```

```
        Dangerous comparisons:
        - block.timestamp < poolStartTime (contracts/distribution/
DracoRewardPool.sol#99)
        - _lastRewardTime == 0 (contracts/distribution/DracoRewardPool.sol#101)
        - _lastRewardTime < poolStartTime (contracts/distribution/
DracoRewardPool.sol#104)
        - _lastRewardTime == 0 || _lastRewardTime < block.timestamp (contracts/
distribution/DracoRewardPool.sol#110)
        - _isStarted = (_lastRewardTime <= poolStartTime) || (_lastRewardTime <=
block.timestamp) (contracts/distribution/DracoRewardPool.sol#114)
DracoRewardPool.getGeneratedReward(uint256,uint256) (contracts/distribution/
DracoRewardPool.sol#132-153) uses timestamp for comparisons
        Dangerous comparisons:
        - _toTime >= epochEndTimes[epochId - 1] (contracts/distribution/
DracoRewardPool.sol#134)
DracoRewardPool.pendingDRACO(uint256,address) (contracts/distribution/
DracoRewardPool.sol#156-167) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > pool.lastRewardTime && tokenSupply != 0 (contracts/
distribution/DracoRewardPool.sol#161)
DracoRewardPool.massUpdatePools() (contracts/distribution/DracoRewardPool.sol#170-175)
uses timestamp for comparisons
        Dangerous comparisons:
        - pid < length (contracts/distribution/DracoRewardPool.sol#172)
DracoRewardPool.updatePool(uint256) (contracts/distribution/
DracoRewardPool.sol#178-198) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp <= pool.lastRewardTime (contracts/distribution/
DracoRewardPool.sol#180)
DracoRewardPool.governanceRecoverUnsupported(IERC20,uint256,address) (contracts/
distribution/DracoRewardPool.sol#268-283) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < epochEndTimes[1] + 2592000 (contracts/distribution/
DracoRewardPool.sol#273)
UniswapV2OracleLibrary.currentCumulativePrices(address) (contracts/lib/
UniswapV2OracleLibrary.sol#18-42) uses timestamp for comparisons
        Dangerous comparisons:
        - blockTimestampLast != blockTimestamp (contracts/lib/
UniswapV2OracleLibrary.sol#33)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp
```

```
INFO:Detectors:
Different versions of Solidity is used:
        - Version used: ['0.6.12', '^0.6.0']
        - 0.6.12 (contracts/Distributor.sol#3)
        - ^0.6.0 (contracts/interfaces/IDistributor.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used
INFO:Detectors:
Draco._updateTaxRate(uint256) (contracts/Draco.sol#127-137) has costly operations
inside a loop:
        - taxRate = taxTiersRates[tierId] (contracts/Draco.sol#132)
Treasury._calculateMaxSupplyExpansionPercent(uint256) (contracts/Treasury.sol#482-490)
has costly operations inside a loop:
        - maxSupplyExpansionPercent = maxExpansionTiers[tierId] (contracts/
Treasury.sol#485)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-
operations-inside-a-loop
INFO:Detectors:
Babylonian.sqrt(uint256) (contracts/lib/Babylonian.sol#6-18) is never used and should
be removed
FixedPoint.decode(FixedPoint.uq112x112) (contracts/lib/FixedPoint.sol#57-59) is never
used and should be removed
FixedPoint.div(FixedPoint.uq112x112,uint112) (contracts/lib/FixedPoint.sol#36-39) is
never used and should be removed
FixedPoint.encode(uint112) (contracts/lib/FixedPoint.sol#26-28) is never used and
should be removed
FixedPoint.encode144(uint144) (contracts/lib/FixedPoint.sol#31-33) is never used and
should be removed
FixedPoint.reciprocal(FixedPoint.uq112x112) (contracts/lib/FixedPoint.sol#67-70) is
never used and should be removed
FixedPoint.sqrt(FixedPoint.uq112x112) (contracts/lib/FixedPoint.sol#73-75) is never
used and should be removed
SafeMath8.add(uint8,uint8) (contracts/lib/SafeMath8.sol#29-34) is never used and should
be removed
SafeMath8.div(uint8,uint8) (contracts/lib/SafeMath8.sol#103-105) is never used and
should be removed
SafeMath8.div(uint8,uint8,string) (contracts/lib/SafeMath8.sol#119-125) is never used
and should be removed
SafeMath8.mod(uint8,uint8) (contracts/lib/SafeMath8.sol#139-141) is never used and
should be removed
SafeMath8.mod(uint8,uint8,string) (contracts/lib/SafeMath8.sol#155-158) is never used
```

```
and should be removed
SafeMath8.mul(uint8,uint8) (contracts/lib/SafeMath8.sol#77-89) is never used and should
be removed
UniswapV2Library.getAmountIn(uint256,uint256,uint256) (contracts/lib/
UniswapV2Library.sol#76-86) is never used and should be removed
UniswapV2Library.getAmountOut(uint256,uint256,uint256) (contracts/lib/
UniswapV2Library.sol#62-73) is never used and should be removed
UniswapV2Library.getAmountsIn(address,uint256,address[]) (contracts/lib/
UniswapV2Library.sol#104-116) is never used and should be removed
UniswapV2Library.getAmountsOut(address,uint256,address[]) (contracts/lib/
UniswapV2Library.sol#89-101) is never used and should be removed
UniswapV2Library.getReserves(address,address,address) (contracts/lib/
UniswapV2Library.sol#40-48) is never used and should be removed
UniswapV2Library.pairFor(address,address,address) (contracts/lib/
UniswapV2Library.sol#19-37) is never used and should be removed
UniswapV2Library.quote(uint256,uint256,uint256) (contracts/lib/
UniswapV2Library.sol#51-59) is never used and should be removed
UniswapV2Library.sortTokens(address,address) (contracts/lib/UniswapV2Library.sol#12-16)
is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Safe112.add(uint112,uint112) (contracts/lib/Safe112.sol#5-10) is never used and should
be removed
Safe112.div(uint112,uint112) (contracts/lib/Safe112.sol#38-40) is never used and should
be removed
Safe112.div(uint112,uint112,string) (contracts/lib/Safe112.sol#42-52) is never used and
should be removed
Safe112.mod(uint112,uint112) (contracts/lib/Safe112.sol#54-56) is never used and should
be removed
Safe112.mod(uint112,uint112,string) (contracts/lib/Safe112.sol#58-65) is never used and
should be removed
Safe112.mul(uint112,uint112) (contracts/lib/Safe112.sol#27-36) is never used and should
be removed
Safe112.sub(uint112,uint112) (contracts/lib/Safe112.sol#12-14) is never used and should
be removed
Safe112.sub(uint112,uint112,string) (contracts/lib/Safe112.sol#16-25) is never used and
should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
UQ112x112.encode(uint112) (contracts/lib/UQ112x112.sol#13-15) is never used and should
be removed
```

```
UQ112x112.uqdiv(uint224,uint112) (contracts/lib/UQ112x112.sol#18-20) is never used and
should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.6.0 (contracts/interfaces/IBasisAsset.sol#2) allows old versions
Pragma version^0.6.0 (contracts/interfaces/IUniswapV2Pair.sol#2) allows old versions
Pragma version^0.6.0 (contracts/lib/Babylonian.sol#3) allows old versions
Pragma version^0.6.0 (contracts/lib/FixedPoint.sol#3) allows old versions
Pragma version^0.6.0 (contracts/lib/UniswapV2Library.sol#3) allows old versions
Pragma version^0.6.0 (contracts/lib/UniswapV2OracleLibrary.sol#3) allows old versions
Pragma version^0.6.0 (contracts/utils/Epoch.sol#3) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Pragma version^0.6.0 (contracts/interfaces/IDistributor.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Pragma version^0.6.0 (contracts/interfaces/ISimpleERCFund.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Pragma version^0.6.0 (contracts/interfaces/IUniswapV2Callee.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Pragma version^0.6.0 (contracts/interfaces/IUniswapV2ERC20.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Pragma version^0.6.0 (contracts/interfaces/IUniswapV2Factory.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Pragma version^0.6.0 (contracts/lib/Safe112.sol#2) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
DBond (contracts/DBond.sol#19-46) should inherit from IBasisAsset (contracts/interfaces/
IBasisAsset.sol#4-16)
Draco (contracts/Draco.sol#21-272) should inherit from IBasisAsset (contracts/
interfaces/IBasisAsset.sol#4-16)
Oracle (contracts/Oracle.sol#24-105) should inherit from IOracle (contracts/interfaces/
IOracle.sol#5-11)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-
inheritance
INFO:Detectors:
Distributor (contracts/Distributor.sol#7-19) should inherit from IDistributor
(contracts/interfaces/IDistributor.sol#4-6)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-
inheritance
INFO:Detectors:
Parameter Draco.isAddressExcluded(address)._address (contracts/Draco.sol#91) is not in
mixedCase
Parameter Draco.setTaxTiersTwap(uint8,uint256)._index (contracts/Draco.sol#95) is not
in mixedCase
Parameter Draco.setTaxTiersTwap(uint8,uint256)._value (contracts/Draco.sol#95) is not
in mixedCase
Parameter Draco.setTaxTiersRate(uint8,uint256)._index (contracts/Draco.sol#108) is not
in mixedCase
Parameter Draco.setTaxTiersRate(uint8,uint256)._value (contracts/Draco.sol#108) is not
in mixedCase
Parameter Draco.setBurnThreshold(uint256)._burnThreshold (contracts/Draco.sol#115) is
not in mixedCase
Parameter Draco.setDracoOracle(address)._dracoOracle (contracts/Draco.sol#147) is not
in mixedCase
Parameter Draco.setTaxOffice(address)._taxOffice (contracts/Draco.sol#152) is not in
mixedCase
Parameter Draco.setTaxCollectorAddress(address)._taxCollectorAddress (contracts/
Draco.sol#158) is not in mixedCase
Parameter Draco.setTaxRate(uint256)._taxRate (contracts/Draco.sol#163) is not in
mixedCase
Parameter Draco.excludeAddress(address)._address (contracts/Draco.sol#169) is not in
mixedCase
Parameter Draco.includeAddress(address)._address (contracts/Draco.sol#175) is not in
mixedCase
Parameter Draco.distributeReward(address)._genesisPool (contracts/Draco.sol#257) is not
in mixedCase
Parameter Draco.governanceRecoverUnsupported(IERC20,uint256,address)._token (contracts/
Draco.sol#266) is not in mixedCase
Parameter Draco.governanceRecoverUnsupported(IERC20,uint256,address)._amount (contracts/
Draco.sol#267) is not in mixedCase
```

```
Parameter Draco.governanceRecoverUnsupported(IERC20,uint256,address)._to (contracts/
Draco.sol#268) is not in mixedCase
Parameter Masonry.initialize(IERC20,IERC20,ITreasury)._draco (contracts/
Masonry.sol#127) is not in mixedCase
Parameter Masonry.initialize(IERC20,IERC20,ITreasury)._share (contracts/
Masonry.sol#128) is not in mixedCase
Parameter Masonry.initialize(IERC20,IERC20,ITreasury)._treasury (contracts/
Masonry.sol#129) is not in mixedCase
Parameter Masonry.setOperator(address)._operator (contracts/Masonry.sol#146) is not in
mixedCase
Parameter Masonry.setLockUp(uint256,uint256)._withdrawLockupEpochs (contracts/
Masonry.sol#150) is not in mixedCase
Parameter Masonry.setLockUp(uint256,uint256)._rewardLockupEpochs (contracts/
Masonry.sol#150) is not in mixedCase
Parameter Masonry.governanceRecoverUnsupported(IERC20,uint256,address)._token
(contracts/Masonry.sol#260) is not in mixedCase
Parameter Masonry.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(contracts/Masonry.sol#260) is not in mixedCase
Parameter Masonry.governanceRecoverUnsupported(IERC20,uint256,address)._to (contracts/
Masonry.sol#260) is not in mixedCase
Parameter Oracle.consult(address,uint256)._token (contracts/Oracle.sol#85) is not in
mixedCase
Parameter Oracle.consult(address,uint256)._amountIn (contracts/Oracle.sol#85) is not in
mixedCase
Parameter Oracle.twap(address,uint256)._token (contracts/Oracle.sol#94) is not in
mixedCase
Parameter Oracle.twap(address,uint256)._amountIn (contracts/Oracle.sol#94) is not in
mixedCase
Parameter SDraco.setTreasuryFund(address)._communityFund (contracts/SDraco.sol#61) is
not in mixedCase
Parameter SDraco.setDevFund(address)._devFund (contracts/SDraco.sol#66) is not in
mixedCase
Parameter SDraco.distributeReward(address)._farmingIncentiveFund (contracts/
SDraco.sol#105) is not in mixedCase
Parameter SDraco.governanceRecoverUnsupported(IERC20,uint256,address)._token (contracts/
SDraco.sol#117) is not in mixedCase
Parameter SDraco.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(contracts/SDraco.sol#118) is not in mixedCase
Parameter SDraco.governanceRecoverUnsupported(IERC20,uint256,address)._to (contracts/
SDraco.sol#119) is not in mixedCase
Parameter SDracoRewardPool.checkPoolDuplicate(IERC20)._token (contracts/
SDracoRewardPool.sol#75) is not in mixedCase
Parameter SDracoRewardPool.add(uint256,IERC20,bool,uint256)._allocPoint (contracts/
SDracoRewardPool.sol#84) is not in mixedCase
Parameter SDracoRewardPool.add(uint256,IERC20,bool,uint256)._token (contracts/
SDracoRewardPool.sol#85) is not in mixedCase
Parameter SDracoRewardPool.add(uint256,IERC20,bool,uint256)._withUpdate (contracts/
SDracoRewardPool.sol#86) is not in mixedCase
Parameter SDracoRewardPool.add(uint256,IERC20,bool,uint256)._lastRewardTime (contracts/
SDracoRewardPool.sol#87) is not in mixedCase
Parameter SDracoRewardPool.set(uint256,uint256)._pid (contracts/
SDracoRewardPool.sol#124) is not in mixedCase
Parameter SDracoRewardPool.set(uint256,uint256)._allocPoint (contracts/
SDracoRewardPool.sol#124) is not in mixedCase
Parameter SDracoRewardPool.getGeneratedReward(uint256,uint256)._fromTime (contracts/
SDracoRewardPool.sol#136) is not in mixedCase
Parameter SDracoRewardPool.getGeneratedReward(uint256,uint256)._toTime (contracts/
SDracoRewardPool.sol#136) is not in mixedCase
Parameter SDracoRewardPool.pendingShare(uint256,address)._pid (contracts/
SDracoRewardPool.sol#150) is not in mixedCase
Parameter SDracoRewardPool.pendingShare(uint256,address)._user (contracts/
SDracoRewardPool.sol#150) is not in mixedCase
Parameter SDracoRewardPool.updatePool(uint256)._pid (contracts/
SDracoRewardPool.sol#172) is not in mixedCase
Parameter SDracoRewardPool.deposit(uint256,uint256)._pid (contracts/
SDracoRewardPool.sol#195) is not in mixedCase
Parameter SDracoRewardPool.deposit(uint256,uint256)._amount (contracts/
SDracoRewardPool.sol#195) is not in mixedCase
Parameter SDracoRewardPool.withdraw(uint256,uint256)._pid (contracts/
SDracoRewardPool.sol#216) is not in mixedCase
Parameter SDracoRewardPool.withdraw(uint256,uint256)._amount (contracts/
SDracoRewardPool.sol#216) is not in mixedCase
Parameter SDracoRewardPool.emergencyWithdraw(uint256)._pid (contracts/
SDracoRewardPool.sol#236) is not in mixedCase
Parameter SDracoRewardPool.safeSDracoTransfer(address,uint256)._to (contracts/
SDracoRewardPool.sol#247) is not in mixedCase
Parameter SDracoRewardPool.safeSDracoTransfer(address,uint256)._amount (contracts/
SDracoRewardPool.sol#247) is not in mixedCase
Parameter SDracoRewardPool.setOperator(address)._operator (contracts/
SDracoRewardPool.sol#258) is not in mixedCase
Parameter SDracoRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token
(contracts/SDracoRewardPool.sol#262) is not in mixedCase
```

```
Parameter TaxOffice.setTaxTiersTwap(uint8,uint256)._index (contracts/TaxOffice.sol#25) is
not in mixedCase
Parameter TaxOffice.setTaxTiersTwap(uint8,uint256)._value (contracts/TaxOffice.sol#25) is
not in mixedCase
Parameter TaxOffice.setTaxTiersRate(uint8,uint256)._index (contracts/TaxOffice.sol#29) is
not in mixedCase
Parameter TaxOffice.setTaxTiersRate(uint8,uint256)._value (contracts/TaxOffice.sol#29) is
not in mixedCase
Parameter TaxOffice.setTaxRate(uint256)._taxRate (contracts/TaxOffice.sol#41) is not in
mixedCase
Parameter TaxOffice.setBurnThreshold(uint256)._burnThreshold (contracts/TaxOffice.sol#45)
is not in mixedCase
Parameter TaxOffice.setTaxCollectorAddress(address)._taxCollectorAddress (contracts/
TaxOffice.sol#49) is not in mixedCase
Parameter TaxOffice.excludeAddressFromTax(address)._address (contracts/TaxOffice.sol#53)
is not in mixedCase
Parameter TaxOffice.includeAddressInTax(address)._address (contracts/TaxOffice.sol#57) is
not in mixedCase
Parameter TaxOffice.setTaxableDracoOracle(address)._dracoOracle (contracts/
TaxOffice.sol#61) is not in mixedCase
Parameter TaxOffice.transferTaxOffice(address)._newTaxOffice (contracts/TaxOffice.sol#65)
is not in mixedCase
Parameter TaxOfficeV2.setTaxTiersTwap(uint8,uint256)._index (contracts/
TaxOfficeV2.sol#36) is not in mixedCase
Parameter TaxOfficeV2.setTaxTiersTwap(uint8,uint256)._value (contracts/
TaxOfficeV2.sol#36) is not in mixedCase
Parameter TaxOfficeV2.setTaxTiersRate(uint8,uint256)._index (contracts/
TaxOfficeV2.sol#40) is not in mixedCase
Parameter TaxOfficeV2.setTaxTiersRate(uint8,uint256)._value (contracts/
TaxOfficeV2.sol#40) is not in mixedCase
Parameter TaxOfficeV2.setTaxRate(uint256)._taxRate (contracts/TaxOfficeV2.sol#52) is not
in mixedCase
Parameter TaxOfficeV2.setBurnThreshold(uint256)._burnThreshold (contracts/
TaxOfficeV2.sol#56) is not in mixedCase
Parameter TaxOfficeV2.setTaxCollectorAddress(address)._taxCollectorAddress (contracts/
TaxOfficeV2.sol#60) is not in mixedCase
Parameter TaxOfficeV2.excludeAddressFromTax(address)._address (contracts/
TaxOfficeV2.sol#64) is not in mixedCase
Parameter TaxOfficeV2.includeAddressInTax(address)._address (contracts/
TaxOfficeV2.sol#74) is not in mixedCase
Parameter TaxOfficeV2.setTaxableDracoOracle(address)._dracoOracle (contracts/
TaxOfficeV2.sol#174) is not in mixedCase
Parameter TaxOfficeV2.transferTaxOffice(address)._newTaxOffice (contracts/
TaxOfficeV2.sol#178) is not in mixedCase
Parameter TaxOfficeV2.taxFreeTransferFrom(address,address,uint256)._sender (contracts/
TaxOfficeV2.sol#183) is not in mixedCase
Parameter TaxOfficeV2.taxFreeTransferFrom(address,address,uint256)._recipient (contracts/
TaxOfficeV2.sol#184) is not in mixedCase
Parameter TaxOfficeV2.taxFreeTransferFrom(address,address,uint256)._amt (contracts/
TaxOfficeV2.sol#185) is not in mixedCase
Parameter TaxOfficeV2.setTaxExclusionForAddress(address,bool)._address (contracts/
TaxOfficeV2.sol#193) is not in mixedCase
Parameter TaxOfficeV2.setTaxExclusionForAddress(address,bool)._excluded (contracts/
TaxOfficeV2.sol#193) is not in mixedCase
Parameter DracoTaxOracle.consult(address)._token (contracts/TaxOracle.sol#38) is not in
mixedCase
Parameter DracoTaxOracle.setDraco(address)._draco (contracts/TaxOracle.sol#45) is not
in mixedCase
Parameter DracoTaxOracle.setWftm(address)._wftm (contracts/TaxOracle.sol#50) is not in
mixedCase
Parameter DracoTaxOracle.setPair(address)._pair (contracts/TaxOracle.sol#55) is not in
mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._draco
(contracts/Treasury.sol#237) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._dbond
(contracts/Treasury.sol#238) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._sdraco
(contracts/Treasury.sol#239) is not in mixedCase
Parameter
Treasury.initialize(address,address,address,address,address,uint256)._dracoOracle
(contracts/Treasury.sol#240) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._masonry
(contracts/Treasury.sol#241) is not in mixedCase
Parameter
Treasury.initialize(address,address,address,address,address,uint256)._startTime
(contracts/Treasury.sol#242) is not in mixedCase
Parameter Treasury.setOperator(address)._operator (contracts/Treasury.sol#280) is not
in mixedCase
Parameter Treasury.setMasonry(address)._masonry (contracts/Treasury.sol#284) is not in
mixedCase
Parameter Treasury.setDracoOracle(address)._dracoOracle (contracts/Treasury.sol#288) is
not in mixedCase
```

```
Parameter Treasury.setDracoPriceCeiling(uint256)._dracoPriceCeiling (contracts/
Treasury.sol#292) is not in mixedCase
Parameter Treasury.setMaxSupplyExpansionPercents(uint256)._maxSupplyExpansionPercent
(contracts/Treasury.sol#297) is not in mixedCase
Parameter Treasury.setSupplyTiersEntry(uint8,uint256)._index (contracts/
Treasury.sol#302) is not in mixedCase
Parameter Treasury.setSupplyTiersEntry(uint8,uint256)._value (contracts/
Treasury.sol#302) is not in mixedCase
Parameter Treasury.setMaxExpansionTiersEntry(uint8,uint256)._index (contracts/
Treasury.sol#315) is not in mixedCase
Parameter Treasury.setMaxExpansionTiersEntry(uint8,uint256)._value (contracts/
Treasury.sol#315) is not in mixedCase
Parameter Treasury.setBondDepletionFloorPercent(uint256)._bondDepletionFloorPercent
(contracts/Treasury.sol#323) is not in mixedCase
Parameter Treasury.setMaxSupplyContractionPercent(uint256)._maxSupplyContractionPercent
(contracts/Treasury.sol#328) is not in mixedCase
Parameter Treasury.setMaxDebtRatioPercent(uint256)._maxDebtRatioPercent (contracts/
Treasury.sol#333) is not in mixedCase
Parameter Treasury.setBootstrap(uint256,uint256)._bootstrapEpochs (contracts/
Treasury.sol#338) is not in mixedCase
Parameter Treasury.setBootstrap(uint256,uint256)._bootstrapSupplyExpansionPercent
(contracts/Treasury.sol#338) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._daoFund (contracts/
Treasury.sol#346) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._daoFundSharedPercent
(contracts/Treasury.sol#347) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._devFund (contracts/
Treasury.sol#348) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._devFundSharedPercent
(contracts/Treasury.sol#349) is not in mixedCase
Parameter Treasury.setMaxDiscountRate(uint256)._maxDiscountRate (contracts/
Treasury.sol#361) is not in mixedCase
Parameter Treasury.setMaxPremiumRate(uint256)._maxPremiumRate (contracts/
Treasury.sol#365) is not in mixedCase
Parameter Treasury.setDiscountPercent(uint256)._discountPercent (contracts/
Treasury.sol#369) is not in mixedCase
Parameter Treasury.setPremiumThreshold(uint256)._premiumThreshold (contracts/
Treasury.sol#374) is not in mixedCase
Parameter Treasury.setPremiumPercent(uint256)._premiumPercent (contracts/
Treasury.sol#380) is not in mixedCase
Parameter Treasury.setMintingFactorForPayingDebt(uint256)._mintingFactorForPayingDebt
```

```
(contracts/Treasury.sol#385) is not in mixedCase
Parameter Treasury.buyBonds(uint256,uint256)._dracoAmount (contracts/Treasury.sol#402)
is not in mixedCase
Parameter Treasury.redeemBonds(uint256,uint256)._bondAmount (contracts/
Treasury.sol#431) is not in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._token
(contracts/Treasury.sol#535) is not in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(contracts/Treasury.sol#536) is not in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._to (contracts/
Treasury.sol#537) is not in mixedCase
Parameter Treasury.masonrySetOperator(address)._operator (contracts/Treasury.sol#546)
is not in mixedCase
Parameter Treasury.masonrySetLockUp(uint256,uint256)._withdrawLockupEpochs (contracts/
Treasury.sol#550) is not in mixedCase
Parameter Treasury.masonrySetLockUp(uint256,uint256)._rewardLockupEpochs (contracts/
Treasury.sol#550) is not in mixedCase
Parameter Treasury.masonryGovernanceRecoverUnsupported(address,uint256,address)._token
(contracts/Treasury.sol#559) is not in mixedCase
Parameter Treasury.masonryGovernanceRecoverUnsupported(address,uint256,address)._amount
(contracts/Treasury.sol#560) is not in mixedCase
Parameter Treasury.masonryGovernanceRecoverUnsupported(address,uint256,address)._to
(contracts/Treasury.sol#561) is not in mixedCase
Parameter DracoGenesisRewardPool.checkPoolDuplicate(IERC20)._token (contracts/
distribution/DracoGenesisRewardPool.sol#82) is not in mixedCase
Parameter DracoGenesisRewardPool.add(uint256,IERC20,bool,uint256)._allocPoint
(contracts/distribution/DracoGenesisRewardPool.sol#94) is not in mixedCase
Parameter DracoGenesisRewardPool.add(uint256,IERC20,bool,uint256)._token (contracts/
distribution/DracoGenesisRewardPool.sol#95) is not in mixedCase
Parameter DracoGenesisRewardPool.add(uint256,IERC20,bool,uint256)._withUpdate
(contracts/distribution/DracoGenesisRewardPool.sol#96) is not in mixedCase
Parameter DracoGenesisRewardPool.add(uint256,IERC20,bool,uint256)._lastRewardTime
(contracts/distribution/DracoGenesisRewardPool.sol#97) is not in mixedCase
Parameter DracoGenesisRewardPool.set(uint256,uint256)._pid (contracts/distribution/
DracoGenesisRewardPool.sol#135) is not in mixedCase
Parameter DracoGenesisRewardPool.set(uint256,uint256)._allocPoint (contracts/
distribution/DracoGenesisRewardPool.sol#135) is not in mixedCase
Parameter DracoGenesisRewardPool.getGeneratedReward(uint256,uint256)._fromTime
(contracts/distribution/DracoGenesisRewardPool.sol#147) is not in mixedCase
Parameter DracoGenesisRewardPool.getGeneratedReward(uint256,uint256)._toTime (contracts/
distribution/DracoGenesisRewardPool.sol#147) is not in mixedCase
```

```
Parameter DracoGenesisRewardPool.pendingDRACO(uint256,address)._pid (contracts/
distribution/DracoGenesisRewardPool.sol#167) is not in mixedCase
Parameter DracoGenesisRewardPool.pendingDRACO(uint256,address)._user (contracts/
distribution/DracoGenesisRewardPool.sol#167) is not in mixedCase
Parameter DracoGenesisRewardPool.updatePool(uint256)._pid (contracts/distribution/
DracoGenesisRewardPool.sol#200) is not in mixedCase
Parameter DracoGenesisRewardPool.deposit(uint256,uint256)._pid (contracts/distribution/
DracoGenesisRewardPool.sol#230) is not in mixedCase
Parameter DracoGenesisRewardPool.deposit(uint256,uint256)._amount (contracts/
distribution/DracoGenesisRewardPool.sol#230) is not in mixedCase
Parameter DracoGenesisRewardPool.withdraw(uint256,uint256)._pid (contracts/distribution/
DracoGenesisRewardPool.sol#261) is not in mixedCase
Parameter DracoGenesisRewardPool.withdraw(uint256,uint256)._amount (contracts/
distribution/DracoGenesisRewardPool.sol#261) is not in mixedCase
Parameter DracoGenesisRewardPool.emergencyWithdraw(uint256)._pid (contracts/
distribution/DracoGenesisRewardPool.sol#283) is not in mixedCase
Parameter DracoGenesisRewardPool.safeDracoTransfer(address,uint256)._to (contracts/
distribution/DracoGenesisRewardPool.sol#294) is not in mixedCase
Parameter DracoGenesisRewardPool.safeDracoTransfer(address,uint256)._amount (contracts/
distribution/DracoGenesisRewardPool.sol#294) is not in mixedCase
Parameter DracoGenesisRewardPool.setOperator(address)._operator (contracts/distribution/
DracoGenesisRewardPool.sol#305) is not in mixedCase
Parameter
DracoGenesisRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token
(contracts/distribution/DracoGenesisRewardPool.sol#310) is not in mixedCase
Parameter DracoRewardPool.checkPoolDuplicate(IERC20)._token (contracts/distribution/
DracoRewardPool.sol#81) is not in mixedCase
Parameter DracoRewardPool.add(uint256,IERC20,bool,uint256)._allocPoint (contracts/
distribution/DracoRewardPool.sol#90) is not in mixedCase
Parameter DracoRewardPool.add(uint256,IERC20,bool,uint256)._token (contracts/
distribution/DracoRewardPool.sol#91) is not in mixedCase
Parameter DracoRewardPool.add(uint256,IERC20,bool,uint256)._withUpdate (contracts/
distribution/DracoRewardPool.sol#92) is not in mixedCase
Parameter DracoRewardPool.add(uint256,IERC20,bool,uint256)._lastRewardTime (contracts/
distribution/DracoRewardPool.sol#93) is not in mixedCase
Parameter DracoRewardPool.set(uint256,uint256)._pid (contracts/distribution/
DracoRewardPool.sol#122) is not in mixedCase
Parameter DracoRewardPool.set(uint256,uint256)._allocPoint (contracts/distribution/
DracoRewardPool.sol#122) is not in mixedCase
Parameter DracoRewardPool.getGeneratedReward(uint256,uint256)._fromTime (contracts/
distribution/DracoRewardPool.sol#132) is not in mixedCase
```

```
Parameter DracoRewardPool.getGeneratedReward(uint256,uint256)._toTime (contracts/
distribution/DracoRewardPool.sol#132) is not in mixedCase
Parameter DracoRewardPool.pendingDRACO(uint256,address)._pid (contracts/distribution/
DracoRewardPool.sol#156) is not in mixedCase
Parameter DracoRewardPool.pendingDRACO(uint256,address)._user (contracts/distribution/
DracoRewardPool.sol#156) is not in mixedCase
Parameter DracoRewardPool.updatePool(uint256)._pid (contracts/distribution/
DracoRewardPool.sol#178) is not in mixedCase
Parameter DracoRewardPool.deposit(uint256,uint256)._pid (contracts/distribution/
DracoRewardPool.sol#201) is not in mixedCase
Parameter DracoRewardPool.deposit(uint256,uint256)._amount (contracts/distribution/
DracoRewardPool.sol#201) is not in mixedCase
Parameter DracoRewardPool.withdraw(uint256,uint256)._pid (contracts/distribution/
DracoRewardPool.sol#222) is not in mixedCase
Parameter DracoRewardPool.withdraw(uint256,uint256)._amount (contracts/distribution/
DracoRewardPool.sol#222) is not in mixedCase
Parameter DracoRewardPool.emergencyWithdraw(uint256)._pid (contracts/distribution/
DracoRewardPool.sol#242) is not in mixedCase
Parameter DracoRewardPool.safeDracoTransfer(address,uint256)._to (contracts/
distribution/DracoRewardPool.sol#253) is not in mixedCase
Parameter DracoRewardPool.safeDracoTransfer(address,uint256)._amount (contracts/
distribution/DracoRewardPool.sol#253) is not in mixedCase
Parameter DracoRewardPool.setOperator(address)._operator (contracts/distribution/
DracoRewardPool.sol#264) is not in mixedCase
Parameter DracoRewardPool.governanceRecoverUnsupported(IERC20,uint256,address)._token
(contracts/distribution/DracoRewardPool.sol#269) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/interfaces/IUniswapV2Pair.sol#30)
is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/interfaces/IUniswapV2Pair.sol#32)
is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/interfaces/
IUniswapV2Pair.sol#51) is not in mixedCase
Function IUniswapV2Router.WETH() (contracts/interfaces/IUniswapV2Router.sol#8) is not
in mixedCase
Struct FixedPoint.uq112x112 (contracts/lib/FixedPoint.sol#11-13) is not in CapWords
Struct FixedPoint.uq144x112 (contracts/lib/FixedPoint.sol#17-19) is not in CapWords
Parameter Epoch.setPeriod(uint256)._period (contracts/utils/Epoch.sol#79) is not in
mixedCase
Parameter Epoch.setEpoch(uint256)._epoch (contracts/utils/Epoch.sol#84) is not in
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
```

```
solidity-naming-conventions
INFO:Detectors:
Function IUniswapV2ERC20.DOMAIN_SEPARATOR() (contracts/interfaces/
IUniswapV2ERC20.sol#30) is not in mixedCase
Function IUniswapV2ERC20.PERMIT_TYPEHASH() (contracts/interfaces/
IUniswapV2ERC20.sol#32) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions
INFO:Detectors:
Variable Oracle.price0Average (contracts/Oracle.sol#39) is too similar to
Oracle.price1Average (contracts/Oracle.sol#40)
Variable Oracle.update().price0Cumulative (contracts/Oracle.sol#64) is too similar to
Oracle.twap(address,uint256).price1Cumulative (contracts/Oracle.sol#95)
Variable Oracle.twap(address,uint256).price0Cumulative (contracts/Oracle.sol#95) is too
similar to Oracle.twap(address,uint256).price1Cumulative (contracts/Oracle.sol#95)
Variable Oracle.update().price0Cumulative (contracts/Oracle.sol#64) is too similar to
Oracle.update().price1Cumulative (contracts/Oracle.sol#64)
Variable Oracle.price0CumulativeLast (contracts/Oracle.sol#37) is too similar to
Oracle.price1CumulativeLast (contracts/Oracle.sol#38)
Variable Oracle.twap(address,uint256).price0Cumulative (contracts/Oracle.sol#95) is too
similar to Oracle.update().price1Cumulative (contracts/Oracle.sol#64)
Variable Treasury.setExtraFunds(address,uint256,address,uint256)._daoFundSharedPercent
(contracts/Treasury.sol#347) is too similar to
Treasury.setExtraFunds(address,uint256,address,uint256)._devFundSharedPercent
(contracts/Treasury.sol#349)
Variable IUniswapV2Router.addLiquidity(address,address,uint256,uint256,uint256,uint256,a
ddress,uint256).amountADesired (contracts/interfaces/IUniswapV2Router.sol#13) is too
similar to IUniswapV2Router.addLiquidity(address,address,uint256,uint256,uint256,uint256
,address,uint256).amountBDesired (contracts/interfaces/IUniswapV2Router.sol#14)
Variable UniswapV2OracleLibrary.currentCumulativePrices(address).price0Cumulative
(contracts/lib/UniswapV2OracleLibrary.sol#22) is too similar to
UniswapV2OracleLibrary.currentCumulativePrices(address).price1Cumulative (contracts/lib/
UniswapV2OracleLibrary.sol#23)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
are-too-similar
INFO:Detectors:
Treasury.initialize(address,address,address,address,address,uint256) (contracts/
Treasury.sol#236-278) uses literals with too many digits:
        - supplyTiers = (0,500000000000000000000000,1000000000000000000000000,1500000000
000000000000000,2000000000000000000000000,5000000000000000000000000,10000000000000000000
000000,20000000000000000000000000,50000000000000000000000000) (contracts/
Treasury.sol#255)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits
INFO:Detectors:
DracoGenesisRewardPool.dracoPerSecond (contracts/distribution/
DracoGenesisRewardPool.sol#52) should be constant
DracoGenesisRewardPool.runningTime (contracts/distribution/
DracoGenesisRewardPool.sol#53) should be constant
DracoGenesisRewardPool.wftm (contracts/distribution/DracoGenesisRewardPool.sol#34)
should be constant
SDracoRewardPool.runningTime (contracts/SDracoRewardPool.sol#51) should be constant
SDracoRewardPool.sDracoPerSecond (contracts/SDracoRewardPool.sol#50) should be constant
TaxOfficeV2.wftm (contracts/TaxOfficeV2.sol#25) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-
variables-that-could-be-declared-constant
INFO:Detectors:
mint(address,uint256) should be declared external:
        - DBond.mint(address,uint256) (contracts/DBond.sol#31-37)
isAddressExcluded(address) should be declared external:
        - Draco.isAddressExcluded(address) (contracts/Draco.sol#91-93)
setTaxTiersTwap(uint8,uint256) should be declared external:
        - Draco.setTaxTiersTwap(uint8,uint256) (contracts/Draco.sol#95-106)
setTaxTiersRate(uint8,uint256) should be declared external:
        - Draco.setTaxTiersRate(uint8,uint256) (contracts/Draco.sol#108-113)
setBurnThreshold(uint256) should be declared external:
        - Draco.setBurnThreshold(uint256) (contracts/Draco.sol#115-117)
enableAutoCalculateTax() should be declared external:
        - Draco.enableAutoCalculateTax() (contracts/Draco.sol#139-141)
disableAutoCalculateTax() should be declared external:
        - Draco.disableAutoCalculateTax() (contracts/Draco.sol#143-145)
setDracoOracle(address) should be declared external:
        - Draco.setDracoOracle(address) (contracts/Draco.sol#147-150)
setTaxOffice(address) should be declared external:
        - Draco.setTaxOffice(address) (contracts/Draco.sol#152-156)
setTaxCollectorAddress(address) should be declared external:
        - Draco.setTaxCollectorAddress(address) (contracts/Draco.sol#158-161)
setTaxRate(uint256) should be declared external:
        - Draco.setTaxRate(uint256) (contracts/Draco.sol#163-167)
includeAddress(address) should be declared external:
        - Draco.includeAddress(address) (contracts/Draco.sol#175-179)
mint(address,uint256) should be declared external:
        - Draco.mint(address,uint256) (contracts/Draco.sol#187-193)
```

```
initialize(IERC20,IERC20,ITreasury) should be declared external:
        - Masonry.initialize(IERC20,IERC20,ITreasury) (contracts/Masonry.sol#126-144)
rewardPerShare() should be declared external:
        - Masonry.rewardPerShare() (contracts/Masonry.sol#198-200)
set(uint256,uint256) should be declared external:
        - SDracoRewardPool.set(uint256,uint256) (contracts/
SDracoRewardPool.sol#124-133)
deposit(uint256,uint256) should be declared external:
        - SDracoRewardPool.deposit(uint256,uint256) (contracts/
SDracoRewardPool.sol#195-213)
withdraw(uint256,uint256) should be declared external:
        - SDracoRewardPool.withdraw(uint256,uint256) (contracts/
SDracoRewardPool.sol#216-233)
emergencyWithdraw(uint256) should be declared external:
        - SDracoRewardPool.emergencyWithdraw(uint256) (contracts/
SDracoRewardPool.sol#236-244)
setTaxTiersTwap(uint8,uint256) should be declared external:
        - TaxOffice.setTaxTiersTwap(uint8,uint256) (contracts/TaxOffice.sol#25-27)
setTaxTiersRate(uint8,uint256) should be declared external:
        - TaxOffice.setTaxTiersRate(uint8,uint256) (contracts/TaxOffice.sol#29-31)
enableAutoCalculateTax() should be declared external:
        - TaxOffice.enableAutoCalculateTax() (contracts/TaxOffice.sol#33-35)
disableAutoCalculateTax() should be declared external:
        - TaxOffice.disableAutoCalculateTax() (contracts/TaxOffice.sol#37-39)
setTaxRate(uint256) should be declared external:
        - TaxOffice.setTaxRate(uint256) (contracts/TaxOffice.sol#41-43)
setBurnThreshold(uint256) should be declared external:
        - TaxOffice.setBurnThreshold(uint256) (contracts/TaxOffice.sol#45-47)
setTaxCollectorAddress(address) should be declared external:
        - TaxOffice.setTaxCollectorAddress(address) (contracts/TaxOffice.sol#49-51)
setTaxTiersTwap(uint8,uint256) should be declared external:
        - TaxOfficeV2.setTaxTiersTwap(uint8,uint256) (contracts/TaxOfficeV2.sol#36-38)
setTaxTiersRate(uint8,uint256) should be declared external:
        - TaxOfficeV2.setTaxTiersRate(uint8,uint256) (contracts/TaxOfficeV2.sol#40-42)
enableAutoCalculateTax() should be declared external:
        - TaxOfficeV2.enableAutoCalculateTax() (contracts/TaxOfficeV2.sol#44-46)
disableAutoCalculateTax() should be declared external:
        - TaxOfficeV2.disableAutoCalculateTax() (contracts/TaxOfficeV2.sol#48-50)
setTaxRate(uint256) should be declared external:
        - TaxOfficeV2.setTaxRate(uint256) (contracts/TaxOfficeV2.sol#52-54)
setBurnThreshold(uint256) should be declared external:
        - TaxOfficeV2.setBurnThreshold(uint256) (contracts/TaxOfficeV2.sol#56-58)
setTaxCollectorAddress(address) should be declared external:
        - TaxOfficeV2.setTaxCollectorAddress(address) (contracts/TaxOfficeV2.sol#60-62)
isInitialized() should be declared external:
        - Treasury.isInitialized() (contracts/Treasury.sol#143-145)
getDracoUpdatedPrice() should be declared external:
        - Treasury.getDracoUpdatedPrice() (contracts/Treasury.sol#161-167)
getReserve() should be declared external:
        - Treasury.getReserve() (contracts/Treasury.sol#170-172)
getBurnableDracoLeft() should be declared external:
        - Treasury.getBurnableDracoLeft() (contracts/Treasury.sol#174-186)
getRedeemableBonds() should be declared external:
        - Treasury.getRedeemableBonds() (contracts/Treasury.sol#188-197)
initialize(address,address,address,address,address,uint256) should be declared
external:
        - Treasury.initialize(address,address,address,address,address,uint256)
(contracts/Treasury.sol#236-278)
set(uint256,uint256) should be declared external:
        - DracoGenesisRewardPool.set(uint256,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#135-144)
deposit(uint256,uint256) should be declared external:
        - DracoGenesisRewardPool.deposit(uint256,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#230-258)
withdraw(uint256,uint256) should be declared external:
        - DracoGenesisRewardPool.withdraw(uint256,uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#261-280)
emergencyWithdraw(uint256) should be declared external:
        - DracoGenesisRewardPool.emergencyWithdraw(uint256) (contracts/distribution/
DracoGenesisRewardPool.sol#283-291)
set(uint256,uint256) should be declared external:
        - DracoRewardPool.set(uint256,uint256) (contracts/distribution/
DracoRewardPool.sol#122-129)
deposit(uint256,uint256) should be declared external:
        - DracoRewardPool.deposit(uint256,uint256) (contracts/distribution/
DracoRewardPool.sol#201-219)
withdraw(uint256,uint256) should be declared external:
        - DracoRewardPool.withdraw(uint256,uint256) (contracts/distribution/
DracoRewardPool.sol#222-239)
emergencyWithdraw(uint256) should be declared external:
        - DracoRewardPool.emergencyWithdraw(uint256) (contracts/distribution/
DracoRewardPool.sol#242-250)
```

```
transferOperator(address) should be declared external:
        - Operator.transferOperator(address) (contracts/owner/Operator.sol#31-33)
getCurrentEpoch() should be declared external:
        - Epoch.getCurrentEpoch() (contracts/utils/Epoch.sol#57-59)
getPeriod() should be declared external:
        - Epoch.getPeriod() (contracts/utils/Epoch.sol#61-63)
getStartTime() should be declared external:
        - Epoch.getStartTime() (contracts/utils/Epoch.sol#65-67)
getLastEpochTime() should be declared external:
        - Epoch.getLastEpochTime() (contracts/utils/Epoch.sol#69-71)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
INFO:Detectors:
distribute() should be declared external:
        - Distributor.distribute() (contracts/Distributor.sol#14-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
INFO:Slither:. analyzed (60 contracts with 75 detectors), 428 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github
integration
```

0x Guard