0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

Magik Finance

February 2022

0xguard.com

hello@0xguard.com

Contents

1. Introduction 3
2. Contracts checked 3
3. Procedure 4
4. Classification of issue severity 4
5. Issues 4
6. Conclusion 6
7. Disclaimer 7
8. Static code analysis results 8

Introduction

This report has been prepared for the Magik 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 Magik Finance are available at the official website: https://magik.finance.

Name	Magik Finance
Audit date	2022-02-03 - 2022-02-03
Language	Solidity
Platform	Fantom Network

Contracts checked

Name	Address
Treasury	0x64e3c1a70e08e769f12b5f554ee9c84e6785644b
MBond	0xca4cdc336fdeb7ee618dd7745bd27758c8e03a91
MSHARERewardPool	0x38f006eb9c6778d02351fbd5966f829e7c4445d7
Masonry	0xac55a55676657d793d965ffa1ccc550b95535634
TaxOffice	0x22956cdae8904e57d47a484cada4aa5c3b327c37
MShare	0xc8ca9026ad0882133ef126824f6852567c571a4e
MAGIK	0x87a5c9b60a3aaf1064006fe64285018e50e0d020

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 Magik Finance Project was compared with the Tomb Project. Magik Finance has changed the implementation of Treasury contract.

The Token contract is affected by a vulnerability that was discovered in the Tomb Project.

In the contract Treasury the array of pools excludedFromTotalSupplywas removed. Also, functions, that are changing the token's operator were added to the contract. This makes it possible to change the taxOffice address, which can set the autoCalculateTax variable. The variable applies taxes upon the transfer.

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

```
Magik.governanceRecoverUnsupported(IERC20,uint256,address) (Magik.sol#1235-1241)
ignores return value by _token.transfer(_to,_amount) (Magik.sol#1240)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

Magik.setTaxTiersTwap(uint8,uint256) (Magik.sol#1059-1070) contains a tautology or
contradiction:
⊠- require(bool,string)(_index >= 0,Index has to be higher than 0) (Magik.sol#1060)
Magik.setTaxTiersRate(uint8,uint256) (Magik.sol#1072-1077) contains a tautology or
contradiction:
⊠- require(bool,string)(_index >= 0,Index has to be higher than 0) (Magik.sol#1073)
Magik._updateTaxRate(uint256) (Magik.sol#1091-1101) contains a tautology or
contradiction:
⊠- tierId >= 0 (Magik.sol#1093)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#tautology-or-
contradiction

Magik._getMagikPrice()._price (Magik.sol#1084) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-
local-variables

Magik._getMagikPrice() (Magik.sol#1083-1089) ignores return value by
IOracle(magikOracle).consult(address(this),1e18) (Magik.sol#1084-1088)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Magik.setBurnThreshold(uint256) (Magik.sol#1079-1081) should emit an event for:
⊠- burnThreshold = _burnThreshold (Magik.sol#1080)
Magik.setTaxRate(uint256) (Magik.sol#1127-1131) should emit an event for:
⊠- taxRate = _taxRate (Magik.sol#1130)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

Variable 'Magik._getMagikPrice()._price (Magik.sol#1084)' in Magik._getMagikPrice()
(Magik.sol#1083-1089) potentially used before declaration: uint256(_price)
(Magik.sol#1085)
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 (Magik.sol#9)
⊠- 0.6.12 (Magik.sol#169)
⊠- >=0.6.0<0.8.0 (Magik.sol#182)
⊠- >=0.6.0<0.8.0 (Magik.sol#399)
⊠- >=0.6.0<0.8.0 (Magik.sol#479)
⊠- >=0.6.0<0.8.0 (Magik.sol#506)
⊠- >=0.6.0<0.8.0 (Magik.sol#576)
⊠- 0.6.12 (Magik.sol#583)
⊠- >=0.6.0<0.8.0 (Magik.sol#624)
⊠- >=0.6.0<0.8.0 (Magik.sol#932)
⊠- 0.6.12 (Magik.sol#975)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used

Magik._updateTaxRate(uint256) (Magik.sol#1091-1101) has costly operations inside a
loop:
⊠- taxRate = taxTiersRates[tierId] (Magik.sol#1096)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-
operations-inside-a-loop

Context._msgData() (Magik.sol#496-499) is never used and should be removed
ERC20._setupDecimals(uint8) (Magik.sol#907-909) is never used and should be removed
SafeMath.div(uint256,uint256,string) (Magik.sol#369-372) is never used and should be
removed
SafeMath.mod(uint256,uint256) (Magik.sol#331-334) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Magik.sol#389-392) is never used and should be
removed
SafeMath.tryAdd(uint256,uint256) (Magik.sol#203-207) is never used and should be
removed
SafeMath.tryDiv(uint256,uint256) (Magik.sol#239-242) is never used and should be
removed
SafeMath.tryMod(uint256,uint256) (Magik.sol#249-252) is never used and should be
removed
SafeMath.tryMul(uint256,uint256) (Magik.sol#224-232) is never used and should be
removed
SafeMath.trySub(uint256,uint256) (Magik.sol#214-217) is never used and should be
removed
SafeMath8.add(uint8,uint8) (Magik.sol#35-40) is never used and should be removed
```

```
SafeMath8.div(uint8,uint8) (Magik.sol#109-111) is never used and should be removed
SafeMath8.div(uint8,uint8,string) (Magik.sol#125-131) is never used and should be
removed
SafeMath8.mod(uint8,uint8) (Magik.sol#145-147) is never used and should be removed
SafeMath8.mod(uint8,uint8,string) (Magik.sol#161-164) is never used and should be
removed
SafeMath8.mul(uint8,uint8) (Magik.sol#83-95) 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 (Magik.sol#9) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#182) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#399) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#479) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#506) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#576) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#624) is too complex
Pragma version>=0.6.0<0.8.0 (Magik.sol#932) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Parameter Magik.isAddressExcluded(address)._address (Magik.sol#1055) is not in
mixedCase
Parameter Magik.setTaxTiersTwap(uint8,uint256)._index (Magik.sol#1059) is not in
mixedCase
Parameter Magik.setTaxTiersTwap(uint8,uint256)._value (Magik.sol#1059) is not in
mixedCase
Parameter Magik.setTaxTiersRate(uint8,uint256)._index (Magik.sol#1072) is not in
mixedCase
Parameter Magik.setTaxTiersRate(uint8,uint256)._value (Magik.sol#1072) is not in
mixedCase
Parameter Magik.setBurnThreshold(uint256)._burnThreshold (Magik.sol#1079) is not in
mixedCase
Parameter Magik.setMagikOracle(address)._magikOracle (Magik.sol#1111) is not in
mixedCase
Parameter Magik.setTaxOffice(address)._taxOffice (Magik.sol#1116) is not in mixedCase
Parameter Magik.setTaxCollectorAddress(address)._taxCollectorAddress (Magik.sol#1122)
is not in mixedCase
Parameter Magik.setTaxRate(uint256)._taxRate (Magik.sol#1127) is not in mixedCase
Parameter Magik.excludeAddress(address)._address (Magik.sol#1133) is not in mixedCase
Parameter Magik.includeAddress(address)._address (Magik.sol#1139) is not in mixedCase
Parameter Magik.distributeReward(address,address,address)._genesisPool (Magik.sol#1221)
```

```
is not in mixedCase
Parameter Magik.distributeReward(address,address,address)._magikPool (Magik.sol#1222)
is not in mixedCase
Parameter Magik.distributeReward(address,address,address)._airdropWallet
(Magik.sol#1223) is not in mixedCase
Parameter Magik.governanceRecoverUnsupported(IERC20,uint256,address)._token
(Magik.sol#1236) is not in mixedCase
Parameter Magik.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(Magik.sol#1237) is not in mixedCase
Parameter Magik.governanceRecoverUnsupported(IERC20,uint256,address)._to
(Magik.sol#1238) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (Magik.sol#497)" inContext (Magik.sol#491-500)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

renounceOwnership() should be declared external:
⊠- Ownable.renounceOwnership() (Magik.sol#556-559)
transferOwnership(address) should be declared external:
⊠- Ownable.transferOwnership(address) (Magik.sol#565-569)
operator() should be declared external:
⊠- Operator.operator() (Magik.sol#597-599)
transferOperator(address) should be declared external:
⊠- Operator.transferOperator(address) (Magik.sol#610-612)
name() should be declared external:
⊠- ERC20.name() (Magik.sol#684-686)
symbol() should be declared external:
⊠- ERC20.symbol() (Magik.sol#692-694)
decimals() should be declared external:
⊠- ERC20.decimals() (Magik.sol#709-711)
totalSupply() should be declared external:
⊠- ERC20.totalSupply() (Magik.sol#716-718)
transfer(address,uint256) should be declared external:
⊠- ERC20.transfer(address,uint256) (Magik.sol#735-738)
approve(address,uint256) should be declared external:
⊠- ERC20.approve(address,uint256) (Magik.sol#754-757)
transferFrom(address,address,uint256) should be declared external:
⊠- ERC20.transferFrom(address,address,uint256) (Magik.sol#772-776)
⊠- Magik.transferFrom(address,address,uint256) (Magik.sol#1167-1192)
```

```
increaseAllowance(address,uint256) should be declared external:
	- ERC20.increaseAllowance(address,uint256) (Magik.sol#790-793)
decreaseAllowance(address,uint256) should be declared external:
	- ERC20.decreaseAllowance(address,uint256) (Magik.sol#809-812)
isAddressExcluded(address) should be declared external:
	- Magik.isAddressExcluded(address) (Magik.sol#1055-1057)
setTaxTiersTwap(uint8,uint256) should be declared external:
	- Magik.setTaxTiersTwap(uint8,uint256) (Magik.sol#1059-1070)
setTaxTiersRate(uint8,uint256) should be declared external:
	- Magik.setTaxTiersRate(uint8,uint256) (Magik.sol#1072-1077)
setBurnThreshold(uint256) should be declared external:
	- Magik.setBurnThreshold(uint256) (Magik.sol#1079-1081)
enableAutoCalculateTax() should be declared external:
	- Magik.enableAutoCalculateTax() (Magik.sol#1103-1105)
disableAutoCalculateTax() should be declared external:
	- Magik.disableAutoCalculateTax() (Magik.sol#1107-1109)
setMagikOracle(address) should be declared external:
	- Magik.setMagikOracle(address) (Magik.sol#1111-1114)
setTaxOffice(address) should be declared external:
	- Magik.setTaxOffice(address) (Magik.sol#1116-1120)
setTaxCollectorAddress(address) should be declared external:
	- Magik.setTaxCollectorAddress(address) (Magik.sol#1122-1125)
setTaxRate(uint256) should be declared external:
	- Magik.setTaxRate(uint256) (Magik.sol#1127-1131)
includeAddress(address) should be declared external:
	- Magik.includeAddress(address) (Magik.sol#1139-1143)
mint(address,uint256) should be declared external:
	- Magik.mint(address,uint256) (Magik.sol#1151-1157)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
./Magik.sol analyzed (10 contracts with 75 detectors), 79 result(s) found
```

```
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1397-1437):
	External calls:
	- _updateMagikPrice() (Treasury.sol#1398)
		- IOracle(magikOracle).update() (Treasury.sol#1304)
	- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
		- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
		- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#880)
```

```
⊠⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠⊠- IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
⊠⊠- IERC20(magik).safeApprove(masonry,0) (Treasury.sol#1381)
⊠⊠- IERC20(magik).safeApprove(masonry,_amount) (Treasury.sol#1382)
⊠⊠- IMasonry(masonry).allocateSeigniorage(_amount) (Treasury.sol#1383)
⊠External calls sending eth:
⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠State variables written after the call(s):
⊠- seigniorageSaved = seigniorageSaved.add(_savedForBond) (Treasury.sol#1431)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities

Treasury._sendToMasonry(uint256) (Treasury.sol#1362-1385) ignores return value by
IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
Treasury._sendToMasonry(uint256) (Treasury.sol#1362-1385) ignores return value by
IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

Treasury.allocateSeigniorage() (Treasury.sol#1397-1437) performs a multiplication on
the result of a division:
⊠-_seigniorage = magikSupply.mul(_percentage).div(1e18) (Treasury.sol#1420)
⊠-_savedForMasonry = _seigniorage.mul(seigniorageExpansionFloorPercent).div(10000)
(Treasury.sol#1421)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply

Reentrancy in Treasury.buyBonds(uint256,uint256) (Treasury.sol#1307-1334):
⊠External calls:
⊠- IBasisAsset(magik).burnFrom(msg.sender,_magikAmount) (Treasury.sol#1327)
⊠- IBasisAsset(tbond).mint(msg.sender,_bondAmount) (Treasury.sol#1328)
⊠State variables written after the call(s):
⊠- epochSupplyContractionLeft = epochSupplyContractionLeft.sub(_magikAmount)
(Treasury.sol#1330)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-1

Treasury.setSupplyTiersEntry(uint8,uint256) (Treasury.sol#1213-1224) contains a
tautology or contradiction:
```

```
⊠- require(bool,string)(_index >= 0,Index has to be higher than 0) (Treasury.sol#1214)
Treasury.setMaxExpansionTiersEntry(uint8,uint256) (Treasury.sol#1226-1232) contains a tautology or contradiction:
⊠- require(bool,string)(_index >= 0,Index has to be higher than 0) (Treasury.sol#1227)
Treasury._calculateMaxSupplyExpansionPercent(uint256) (Treasury.sol#1387-1395) contains a tautology or contradiction:
⊠- tierId >= 0 (Treasury.sol#1388)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#tautology-or-contradiction

Treasury.getMagikUpdatedPrice().price (Treasury.sol#1073) is a local variable never initialized
Treasury.getMagikPrice().price (Treasury.sol#1065) is a local variable never initialized
Treasury.allocateSeigniorage()._savedForBond (Treasury.sol#1409) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

Treasury.getMagikPrice() (Treasury.sol#1064-1070) ignores return value by IOracle(magikOracle).consult(magik,1e18) (Treasury.sol#1065-1069)
Treasury.getMagikUpdatedPrice() (Treasury.sol#1072-1078) ignores return value by IOracle(magikOracle).twap(magik,1e18) (Treasury.sol#1073-1077)
Treasury.buyBonds(uint256,uint256) (Treasury.sol#1307-1334) ignores return value by IBasisAsset(tbond).mint(msg.sender,_bondAmount) (Treasury.sol#1328)
Treasury._sendToMasonry(uint256) (Treasury.sol#1362-1385) ignores return value by IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
Treasury.allocateSeigniorage() (Treasury.sol#1397-1437) ignores return value by IBasisAsset(magik).mint(address(this),_savedForBond) (Treasury.sol#1432)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Treasury.setOperator(address) (Treasury.sol#1191-1193) should emit an event for:
⊠- operator = _operator (Treasury.sol#1192)
Treasury.setMasonry(address) (Treasury.sol#1195-1197) should emit an event for:
⊠- masonry = _masonry (Treasury.sol#1196)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

Treasury.setMagikPriceCeiling(uint256) (Treasury.sol#1203-1206) should emit an event for:
⊠- magikPriceCeiling = _magikPriceCeiling (Treasury.sol#1205)
```

```
Treasury.setMaxSupplyExpansionPercents(uint256) (Treasury.sol#1208-1211) should emit an
event for:
⊠- maxSupplyExpansionPercent = _maxSupplyExpansionPercent (Treasury.sol#1210)
Treasury.setBondDepletionFloorPercent(uint256) (Treasury.sol#1234-1237) should emit an
event for:
⊠- bondDepletionFloorPercent = _bondDepletionFloorPercent (Treasury.sol#1236)
Treasury.setMaxDebtRatioPercent(uint256) (Treasury.sol#1244-1247) should emit an event
for:
⊠- maxDebtRatioPercent = _maxDebtRatioPercent (Treasury.sol#1246)
Treasury.setBootstrap(uint256,uint256) (Treasury.sol#1249-1254) should emit an event
for:
⊠- bootstrapEpochs = _bootstrapEpochs (Treasury.sol#1252)
⊠- bootstrapSupplyExpansionPercent = _bootstrapSupplyExpansionPercent
(Treasury.sol#1253)
Treasury.setExtraFunds(address,uint256,address,uint256) (Treasury.sol#1256-1270) should
emit an event for:
⊠- daoFundSharedPercent = _daoFundSharedPercent (Treasury.sol#1267)
⊠- devFundSharedPercent = _devFundSharedPercent (Treasury.sol#1269)
Treasury.setMaxDiscountRate(uint256) (Treasury.sol#1272-1274) should emit an event for:
⊠- maxDiscountRate = _maxDiscountRate (Treasury.sol#1273)
Treasury.setMaxPremiumRate(uint256) (Treasury.sol#1276-1278) should emit an event for:
⊠- maxPremiumRate = _maxPremiumRate (Treasury.sol#1277)
Treasury.setDiscountPercent(uint256) (Treasury.sol#1280-1283) should emit an event for:
⊠- discountPercent = _discountPercent (Treasury.sol#1282)
Treasury.setPremiumThreshold(uint256) (Treasury.sol#1285-1289) should emit an event
for:
⊠- premiumThreshold = _premiumThreshold (Treasury.sol#1288)
Treasury.setPremiumPercent(uint256) (Treasury.sol#1291-1294) should emit an event for:
⊠- premiumPercent = _premiumPercent (Treasury.sol#1293)
Treasury.setMintingFactorForPayingDebt(uint256) (Treasury.sol#1296-1299) should emit an
event for:
⊠- mintingFactorForPayingDebt = _mintingFactorForPayingDebt (Treasury.sol#1298)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

Treasury.initialize(address,address,address,address,address,uint256)._magik
(Treasury.sol#1148) lacks a zero-check on :
⊠⊠- magik = _magik (Treasury.sol#1155)
Treasury.initialize(address,address,address,address,address,uint256)._tbond
(Treasury.sol#1149) lacks a zero-check on :
⊠⊠- tbond = _tbond (Treasury.sol#1156)
```

```
Treasury.initialize(address,address,address,address,address,uint256)._tshare
(Treasury.sol#1150) lacks a zero-check on :
⊠⊠- tshare = _tshare (Treasury.sol#1157)
Treasury.initialize(address,address,address,address,address,uint256)._magikOracle
(Treasury.sol#1151) lacks a zero-check on :
⊠⊠- magikOracle = _magikOracle (Treasury.sol#1158)
Treasury.initialize(address,address,address,address,address,uint256)._masonry
(Treasury.sol#1152) lacks a zero-check on :
⊠⊠- masonry = _masonry (Treasury.sol#1159)
Treasury.setOperator(address)._operator (Treasury.sol#1191) lacks a zero-check on :
⊠⊠- operator = _operator (Treasury.sol#1192)
Treasury.setMasonry(address)._masonry (Treasury.sol#1195) lacks a zero-check on :
⊠⊠- masonry = _masonry (Treasury.sol#1196)
Treasury.setMagikOracle(address)._magikOracle (Treasury.sol#1199) lacks a zero-check
on :
⊠⊠- magikOracle = _magikOracle (Treasury.sol#1200)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

Variable 'Treasury.getMagikPrice().price (Treasury.sol#1065)' in
Treasury.getMagikPrice() (Treasury.sol#1064-1070) potentially used before declaration:
uint256(price) (Treasury.sol#1066)
Variable 'Treasury.getMagikUpdatedPrice().price (Treasury.sol#1073)' in
Treasury.getMagikUpdatedPrice() (Treasury.sol#1072-1078) potentially used before
declaration: uint256(price) (Treasury.sol#1074)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-
declaration-usage-of-local-variables

Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1397-1437):
⊠External calls:
⊠- _updateMagikPrice() (Treasury.sol#1398)
⊠⊠- IOracle(magikOracle).update() (Treasury.sol#1304)
⊠State variables written after the call(s):
⊠- _mse = _calculateMaxSupplyExpansionPercent(magikSupply).mul(1e14)
(Treasury.sol#1411)
⊠⊠- maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1390)
⊠- previousEpochMagikPrice = getMagikPrice() (Treasury.sol#1399)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-2

Reentrancy in Treasury._sendToMasonry(uint256) (Treasury.sol#1362-1385):
```

```
⊠External calls:
⊠- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠Event emitted after the call(s):
⊠- DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1369)
Reentrancy in Treasury._sendToMasonry(uint256) (Treasury.sol#1362-1385):
⊠External calls:
⊠- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠- IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
⊠Event emitted after the call(s):
⊠- DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1376)
Reentrancy in Treasury._sendToMasonry(uint256) (Treasury.sol#1362-1385):
⊠External calls:
⊠- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠- IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
⊠- IERC20(magik).safeApprove(masonry,0) (Treasury.sol#1381)
⊠- IERC20(magik).safeApprove(masonry,_amount) (Treasury.sol#1382)
⊠- IMasonry(masonry).allocateSeigniorage(_amount) (Treasury.sol#1383)
⊠Event emitted after the call(s):
⊠- MasonryFunded(now,_amount) (Treasury.sol#1384)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1397-1437):
⊠External calls:
⊠- _updateMagikPrice() (Treasury.sol#1398)
⊠⊠- IOracle(magikOracle).update() (Treasury.sol#1304)
⊠- _sendToMasonry(magikSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1403)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#880)
⊠⊠- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
⊠⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠⊠- IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
⊠⊠- IERC20(magik).safeApprove(masonry,0) (Treasury.sol#1381)
⊠⊠- IERC20(magik).safeApprove(masonry,_amount) (Treasury.sol#1382)
⊠⊠- IMasonry(masonry).allocateSeigniorage(_amount) (Treasury.sol#1383)
⊠External calls sending eth:
⊠- _sendToMasonry(magikSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1403)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
```

```
⊠Event emitted after the call(s):
⊠- DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1369)
⊠⊠- _sendToMasonry(magikSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1403)
⊠- DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1376)
⊠⊠- _sendToMasonry(magikSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1403)
⊠- MasonryFunded(now,_amount) (Treasury.sol#1384)
⊠⊠- _sendToMasonry(magikSupply.mul(bootstrapSupplyExpansionPercent).div(10000))
(Treasury.sol#1403)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1397-1437):
⊠External calls:
⊠- _updateMagikPrice() (Treasury.sol#1398)
⊠⊠- IOracle(magikOracle).update() (Treasury.sol#1304)
⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#880)
⊠⊠- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
⊠⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠⊠- IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
⊠⊠- IERC20(magik).safeApprove(masonry,0) (Treasury.sol#1381)
⊠⊠- IERC20(magik).safeApprove(masonry,_amount) (Treasury.sol#1382)
⊠⊠- IMasonry(masonry).allocateSeigniorage(_amount) (Treasury.sol#1383)
⊠External calls sending eth:
⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠Event emitted after the call(s):
⊠- DaoFundFunded(now,_daoFundSharedAmount) (Treasury.sol#1369)
⊠⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠- DevFundFunded(now,_devFundSharedAmount) (Treasury.sol#1376)
⊠⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠- MasonryFunded(now,_amount) (Treasury.sol#1384)
⊠⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
Reentrancy in Treasury.allocateSeigniorage() (Treasury.sol#1397-1437):
⊠External calls:
⊠- _updateMagikPrice() (Treasury.sol#1398)
⊠⊠- IOracle(magikOracle).update() (Treasury.sol#1304)
⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠⊠- IBasisAsset(magik).mint(address(this),_amount) (Treasury.sol#1363)
⊠⊠- returndata = address(token).functionCall(data,SafeERC20: low-level call failed)
(Treasury.sol#880)
```

```
⊠⊠- IERC20(magik).transfer(daoFund,_daoFundSharedAmount) (Treasury.sol#1368)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠⊠- IERC20(magik).transfer(devFund,_devFundSharedAmount) (Treasury.sol#1375)
⊠⊠- IERC20(magik).safeApprove(masonry,0) (Treasury.sol#1381)
⊠⊠- IERC20(magik).safeApprove(masonry,_amount) (Treasury.sol#1382)
⊠⊠- IMasonry(masonry).allocateSeigniorage(_amount) (Treasury.sol#1383)
⊠- IBasisAsset(magik).mint(address(this),_savedForBond) (Treasury.sol#1432)
⊠External calls sending eth:
⊠- _sendToMasonry(_savedForMasonry) (Treasury.sol#1428)
⊠⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
⊠Event emitted after the call(s):
⊠- TreasuryFunded(now,_savedForBond) (Treasury.sol#1433)
Reentrancy in Treasury.buyBonds(uint256,uint256) (Treasury.sol#1307-1334):
⊠External calls:
⊠- IBasisAsset(magik).burnFrom(msg.sender,_magikAmount) (Treasury.sol#1327)
⊠- IBasisAsset(tbond).mint(msg.sender,_bondAmount) (Treasury.sol#1328)
⊠- _updateMagikPrice() (Treasury.sol#1331)
⊠⊠- IOracle(magikOracle).update() (Treasury.sol#1304)
⊠Event emitted after the call(s):
⊠- BoughtBonds(msg.sender,_magikAmount,_bondAmount) (Treasury.sol#1333)
Reentrancy in Treasury.redeemBonds(uint256,uint256) (Treasury.sol#1336-1360):
⊠External calls:
⊠- IBasisAsset(tbond).burnFrom(msg.sender,_bondAmount) (Treasury.sol#1354)
⊠- IERC20(magik).safeTransfer(msg.sender,_magikAmount) (Treasury.sol#1355)
⊠- _updateMagikPrice() (Treasury.sol#1357)
⊠⊠- IOracle(magikOracle).update() (Treasury.sol#1304)
⊠Event emitted after the call(s):
⊠- RedeemedBonds(msg.sender,_magikAmount,_bondAmount) (Treasury.sol#1359)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3

Address.isContract(address) (Treasury.sol#349-358) uses assembly
⊠- INLINE ASM (Treasury.sol#356)
Address._verifyCallResult(bool,bytes,string) (Treasury.sol#494-511) uses assembly
⊠- INLINE ASM (Treasury.sol#503-506)
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']
⊠- 0.6.12 (Treasury.sol#4)
⊠- 0.6.12 (Treasury.sol#40)
⊠- ^0.6.0 (Treasury.sol#51)
⊠- ^0.6.0 (Treasury.sol#69)
⊠- >=0.6.0<0.8.0 (Treasury.sol#96)
⊠- >=0.6.0<0.8.0 (Treasury.sol#123)
⊠- >=0.6.0<0.8.0 (Treasury.sol#193)
⊠- 0.6.12 (Treasury.sol#200)
⊠- ^0.6.0 (Treasury.sol#240)
⊠- >=0.6.0<0.8.0 (Treasury.sol#261)
⊠- >=0.6.2<0.8.0 (Treasury.sol#326)
⊠- >=0.6.0<0.8.0 (Treasury.sol#518)
⊠- >=0.6.0<0.8.0 (Treasury.sol#735)
⊠- >=0.6.0<0.8.0 (Treasury.sol#815)
⊠- >=0.6.0<0.8.0 (Treasury.sol#892)
⊠- 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#1387-1395) has
costly operations inside a loop:
⊠- maxSupplyExpansionPercent = maxExpansionTiers[tierId] (Treasury.sol#1390)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-
operations-inside-a-loop

Address.functionCall(address,bytes) (Treasury.sol#402-404) is never used and should be
removed
Address.functionCallWithValue(address,bytes,uint256) (Treasury.sol#427-429) is never
used and should be removed
Address.functionDelegateCall(address,bytes) (Treasury.sol#476-478) is never used and
should be removed
Address.functionDelegateCall(address,bytes,string) (Treasury.sol#486-492) is never used
and should be removed
Address.functionStaticCall(address,bytes) (Treasury.sol#452-454) is never used and
should be removed
Address.functionStaticCall(address,bytes,string) (Treasury.sol#462-468) is never used
and should be removed
Address.sendValue(address,uint256) (Treasury.sol#376-382) is never used and should be
removed
Babylonian.sqrt(uint256) (Treasury.sol#243-255) is never used and should be removed
Context._msgData() (Treasury.sol#113-116) is never used and should be removed
Math.average(uint256,uint256) (Treasury.sol#916-919) is never used and should be
removed
```

```
Math.max(uint256,uint256) (Treasury.sol#901-903) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (Treasury.sol#864-867) is never
used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (Treasury.sol#859-862) is never
used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (Treasury.sol#837-839) is
never used and should be removed
SafeMath.div(uint256,uint256,string) (Treasury.sol#705-708) is never used and should be
removed
SafeMath.mod(uint256,uint256) (Treasury.sol#667-670) is never used and should be
removed
SafeMath.mod(uint256,uint256,string) (Treasury.sol#725-728) is never used and should be
removed
SafeMath.sub(uint256,uint256,string) (Treasury.sol#685-688) is never used and should be
removed
SafeMath.tryAdd(uint256,uint256) (Treasury.sol#539-543) is never used and should be
removed
SafeMath.tryDiv(uint256,uint256) (Treasury.sol#575-578) is never used and should be
removed
SafeMath.tryMod(uint256,uint256) (Treasury.sol#585-588) is never used and should be
removed
SafeMath.tryMul(uint256,uint256) (Treasury.sol#560-568) is never used and should be
removed
SafeMath.trySub(uint256,uint256) (Treasury.sol#550-553) is never used and should be
removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.6.0 (Treasury.sol#51) allows old versions
Pragma version^0.6.0 (Treasury.sol#69) allows old versions
Pragma version>=0.6.0<0.8.0 (Treasury.sol#96) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#123) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#193) is too complex
Pragma version^0.6.0 (Treasury.sol#240) allows old versions
Pragma version>=0.6.0<0.8.0 (Treasury.sol#261) is too complex
Pragma version>=0.6.2<0.8.0 (Treasury.sol#326) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#518) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#735) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#815) is too complex
Pragma version>=0.6.0<0.8.0 (Treasury.sol#892) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
```

```
Low level call in Address.sendValue(address,uint256) (Treasury.sol#376-382):
⊠- (success) = recipient.call{value: amount}() (Treasury.sol#380)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string)
(Treasury.sol#437-444):
⊠- (success,returndata) = target.call{value: value}(data) (Treasury.sol#442)
Low level call in Address.functionStaticCall(address,bytes,string)
(Treasury.sol#462-468):
⊠- (success,returndata) = target.staticcall(data) (Treasury.sol#466)
Low level call in Address.functionDelegateCall(address,bytes,string)
(Treasury.sol#486-492):
⊠- (success,returndata) = target.delegatecall(data) (Treasury.sol#490)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-
calls

Parameter Treasury.initialize(address,address,address,address,address,uint256)._magik
(Treasury.sol#1148) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._tbond
(Treasury.sol#1149) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._tshare
(Treasury.sol#1150) is not in mixedCase
Parameter
Treasury.initialize(address,address,address,address,address,uint256)._magikOracle
(Treasury.sol#1151) is not in mixedCase
Parameter Treasury.initialize(address,address,address,address,address,uint256)._masonry
(Treasury.sol#1152) is not in mixedCase
Parameter
Treasury.initialize(address,address,address,address,address,uint256)._startTime
(Treasury.sol#1153) is not in mixedCase
Parameter Treasury.setOperator(address)._operator (Treasury.sol#1191) is not in
mixedCase
Parameter Treasury.setMasonry(address)._masonry (Treasury.sol#1195) is not in mixedCase
Parameter Treasury.setMagikOracle(address)._magikOracle (Treasury.sol#1199) is not in
mixedCase
Parameter Treasury.setMagikPriceCeiling(uint256)._magikPriceCeiling (Treasury.sol#1203)
is not in mixedCase
Parameter Treasury.setMaxSupplyExpansionPercents(uint256)._maxSupplyExpansionPercent
(Treasury.sol#1208) is not in mixedCase
Parameter Treasury.setSupplyTiersEntry(uint8,uint256)._index (Treasury.sol#1213) is not
in mixedCase
Parameter Treasury.setSupplyTiersEntry(uint8,uint256)._value (Treasury.sol#1213) is not
```

```
in mixedCase
Parameter Treasury.setMaxExpansionTiersEntry(uint8,uint256)._index (Treasury.sol#1226)
is not in mixedCase
Parameter Treasury.setMaxExpansionTiersEntry(uint8,uint256)._value (Treasury.sol#1226)
is not in mixedCase
Parameter Treasury.setBondDepletionFloorPercent(uint256)._bondDepletionFloorPercent
(Treasury.sol#1234) is not in mixedCase
Parameter Treasury.setMaxSupplyContractionPercent(uint256)._maxSupplyContractionPercent
(Treasury.sol#1239) is not in mixedCase
Parameter Treasury.setMaxDebtRatioPercent(uint256)._maxDebtRatioPercent
(Treasury.sol#1244) is not in mixedCase
Parameter Treasury.setBootstrap(uint256,uint256)._bootstrapEpochs (Treasury.sol#1249)
is not in mixedCase
Parameter Treasury.setBootstrap(uint256,uint256)._bootstrapSupplyExpansionPercent
(Treasury.sol#1249) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._daoFund
(Treasury.sol#1257) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._daoFundSharedPercent
(Treasury.sol#1258) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._devFund
(Treasury.sol#1259) is not in mixedCase
Parameter Treasury.setExtraFunds(address,uint256,address,uint256)._devFundSharedPercent
(Treasury.sol#1260) is not in mixedCase
Parameter Treasury.setMaxDiscountRate(uint256)._maxDiscountRate (Treasury.sol#1272) is
not in mixedCase
Parameter Treasury.setMaxPremiumRate(uint256)._maxPremiumRate (Treasury.sol#1276) is
not in mixedCase
Parameter Treasury.setDiscountPercent(uint256)._discountPercent (Treasury.sol#1280) is
not in mixedCase
Parameter Treasury.setPremiumThreshold(uint256)._premiumThreshold (Treasury.sol#1285)
is not in mixedCase
Parameter Treasury.setPremiumPercent(uint256)._premiumPercent (Treasury.sol#1291) is
not in mixedCase
Parameter Treasury.setMintingFactorForPayingDebt(uint256)._mintingFactorForPayingDebt
(Treasury.sol#1296) is not in mixedCase
Parameter Treasury.buyBonds(uint256,uint256)._magikAmount (Treasury.sol#1307) is not in
mixedCase
Parameter Treasury.redeemBonds(uint256,uint256)._bondAmount (Treasury.sol#1336) is not
in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._token
(Treasury.sol#1440) is not in mixedCase
```

```
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(Treasury.sol#1441) is not in mixedCase
Parameter Treasury.governanceRecoverUnsupported(IERC20,uint256,address)._to
(Treasury.sol#1442) is not in mixedCase
Parameter Treasury.magikSetOperator(address)._operator (Treasury.sol#1451) is not in
mixedCase
Parameter Treasury.tshareSetOperator(address)._operator (Treasury.sol#1455) is not in
mixedCase
Parameter Treasury.tbondSetOperator(address)._operator (Treasury.sol#1459) is not in
mixedCase
Parameter Treasury.masonrySetOperator(address)._operator (Treasury.sol#1463) is not in
mixedCase
Parameter Treasury.masonrySetLockUp(uint256,uint256)._withdrawLockupEpochs
(Treasury.sol#1467) is not in mixedCase
Parameter Treasury.masonrySetLockUp(uint256,uint256)._rewardLockupEpochs
(Treasury.sol#1467) is not in mixedCase
Parameter Treasury.masonryGovernanceRecoverUnsupported(address,uint256,address)._token
(Treasury.sol#1476) is not in mixedCase
Parameter Treasury.masonryGovernanceRecoverUnsupported(address,uint256,address)._amount
(Treasury.sol#1477) is not in mixedCase
Parameter Treasury.masonryGovernanceRecoverUnsupported(address,uint256,address)._to
(Treasury.sol#1478) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (Treasury.sol#114)" inContext (Treasury.sol#108-117)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

Variable Treasury.setExtraFunds(address,uint256,address,uint256)._daoFundSharedPercent
(Treasury.sol#1258) is too similar to
Treasury.setExtraFunds(address,uint256,address,uint256)._devFundSharedPercent
(Treasury.sol#1260)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
are-too-similar

Treasury.initialize(address,address,address,address,address,uint256)
(Treasury.sol#1147-1189) uses literals with too many digits:
⊠- supplyTiers = (0,500000000000000000000000,1000000000000000000000000,15000000000000000
00000000,2000000000000000000000000,5000000000000000000000000,10000000000000000000000000,
20000000000000000000000000,50000000000000000000000000) (Treasury.sol#1166)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits

renounceOwnership() should be declared external:
⊠- Ownable.renounceOwnership() (Treasury.sol#173-176)
transferOwnership(address) should be declared external:
⊠- Ownable.transferOwnership(address) (Treasury.sol#182-186)
operator() should be declared external:
⊠- Operator.operator() (Treasury.sol#214-216)
isOperator() should be declared external:
⊠- Operator.isOperator() (Treasury.sol#223-225)
transferOperator(address) should be declared external:
⊠- Operator.transferOperator(address) (Treasury.sol#227-229)
isInitialized() should be declared external:
⊠- Treasury.isInitialized() (Treasury.sol#1054-1056)
getMagikUpdatedPrice() should be declared external:
⊠- Treasury.getMagikUpdatedPrice() (Treasury.sol#1072-1078)
getReserve() should be declared external:
⊠- Treasury.getReserve() (Treasury.sol#1081-1083)
getBurnableMagikLeft() should be declared external:
⊠- Treasury.getBurnableMagikLeft() (Treasury.sol#1085-1097)
getRedeemableBonds() should be declared external:
⊠- Treasury.getRedeemableBonds() (Treasury.sol#1099-1108)
initialize(address,address,address,address,address,uint256) should be declared
external:
⊠- Treasury.initialize(address,address,address,address,address,uint256)
(Treasury.sol#1147-1189)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
```

```
MShare.governanceRecoverUnsupported(IERC20,uint256,address) (MShare.sol#900-906)
ignores return value by _token.transfer(_to,_amount) (MShare.sol#905)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

MShare.setTreasuryFund(address)._communityFund (MShare.sol#845) lacks a zero-check on :
⊠⊠- communityFund = _communityFund (MShare.sol#847)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation
```

```
MShare.unclaimedTreasuryFund() (MShare.sol#856-861) uses timestamp for comparisons
	Dangerous comparisons:
	- _now > endTime (MShare.sol#858)
	- communityFundLastClaimed >= _now (MShare.sol#859)
MShare.unclaimedDevFund() (MShare.sol#863-868) uses timestamp for comparisons
	Dangerous comparisons:
	- _now > endTime (MShare.sol#865)
	- devFundLastClaimed >= _now (MShare.sol#866)
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 (MShare.sol#5)
	- >=0.6.0<0.8.0 (MShare.sol#85)
	- >=0.6.0<0.8.0 (MShare.sol#112)
	- >=0.6.0<0.8.0 (MShare.sol#182)
	- 0.6.12 (MShare.sol#189)
	- >=0.6.0<0.8.0 (MShare.sol#230)
	- >=0.6.0<0.8.0 (MShare.sol#447)
	- >=0.6.0<0.8.0 (MShare.sol#755)
	- 0.6.12 (MShare.sol#798)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Context._msgData() (MShare.sol#102-105) is never used and should be removed
ERC20._setupDecimals(uint8) (MShare.sol#730-732) is never used and should be removed
SafeMath.div(uint256,uint256,string) (MShare.sol#417-420) is never used and should be removed
SafeMath.mod(uint256,uint256) (MShare.sol#379-382) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (MShare.sol#437-440) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (MShare.sol#251-255) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (MShare.sol#287-290) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (MShare.sol#297-300) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (MShare.sol#272-280) is never used and should be removed
SafeMath.trySub(uint256,uint256) (MShare.sol#262-265) 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 (MShare.sol#5) is too complex
Pragma version>=0.6.0<0.8.0 (MShare.sol#85) is too complex
Pragma version>=0.6.0<0.8.0 (MShare.sol#112) is too complex
Pragma version>=0.6.0<0.8.0 (MShare.sol#182) is too complex
Pragma version>=0.6.0<0.8.0 (MShare.sol#230) is too complex
Pragma version>=0.6.0<0.8.0 (MShare.sol#447) is too complex
Pragma version>=0.6.0<0.8.0 (MShare.sol#755) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Parameter MShare.setTreasuryFund(address)._communityFund (MShare.sol#845) is not in
mixedCase
Parameter MShare.setDevFund(address)._devFund (MShare.sol#850) is not in mixedCase
Parameter MShare.distributeReward(address)._farmingIncentiveFund (MShare.sol#889) is
not in mixedCase
Parameter MShare.governanceRecoverUnsupported(IERC20,uint256,address)._token
(MShare.sol#901) is not in mixedCase
Parameter MShare.governanceRecoverUnsupported(IERC20,uint256,address)._amount
(MShare.sol#902) is not in mixedCase
Parameter MShare.governanceRecoverUnsupported(IERC20,uint256,address)._to
(MShare.sol#903) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (MShare.sol#103)" inContext (MShare.sol#97-106)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

renounceOwnership() should be declared external:
⊠- Ownable.renounceOwnership() (MShare.sol#162-165)
transferOwnership(address) should be declared external:
⊠- Ownable.transferOwnership(address) (MShare.sol#171-175)
operator() should be declared external:
⊠- Operator.operator() (MShare.sol#203-205)
isOperator() should be declared external:
⊠- Operator.isOperator() (MShare.sol#212-214)
transferOperator(address) should be declared external:
⊠- Operator.transferOperator(address) (MShare.sol#216-218)
name() should be declared external:
```

```
⊠- ERC20.name() (MShare.sol#507-509)
symbol() should be declared external:
⊠- ERC20.symbol() (MShare.sol#515-517)
decimals() should be declared external:
⊠- ERC20.decimals() (MShare.sol#532-534)
totalSupply() should be declared external:
⊠- ERC20.totalSupply() (MShare.sol#539-541)
balanceOf(address) should be declared external:
⊠- ERC20.balanceOf(address) (MShare.sol#546-548)
transfer(address,uint256) should be declared external:
⊠- ERC20.transfer(address,uint256) (MShare.sol#558-561)
approve(address,uint256) should be declared external:
⊠- ERC20.approve(address,uint256) (MShare.sol#577-580)
transferFrom(address,address,uint256) should be declared external:
⊠- ERC20.transferFrom(address,address,uint256) (MShare.sol#595-599)
increaseAllowance(address,uint256) should be declared external:
⊠- ERC20.increaseAllowance(address,uint256) (MShare.sol#613-616)
decreaseAllowance(address,uint256) should be declared external:
⊠- ERC20.decreaseAllowance(address,uint256) (MShare.sol#632-635)
burnFrom(address,uint256) should be declared external:
⊠- ERC20Burnable.burnFrom(address,uint256) (MShare.sol#787-792)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
./MShare.sol analyzed (8 contracts with 75 detectors), 45 result(s) found
```

0x Guard