
# ZEROSEC AUDIT REPORT

## PROJECT NAME: MASTER KEY FINANCE

## CONTRACT: TOKENWITHFLEX.SOL

## WEBSITE: HTTPS://MASTERKEY.FINANCE

# EXECUTIVE SUMMARY

| Impact Level | Definition |
|---|---|
| Ad Hoc/Automated/High | The issue has a high impact on the contract's security and functionality. |
| Ad Hoc/Automated/Medium | The issue has a medium impact on the contract's security and functionality. |
| Ad Hoc/Automated/Low | The issue has a low impact on the contract's security and functionality. |
| Informational | The issue provides informational details but does not affect security or functionality. |
| Optimization | The issue relates to code optimization and does not affect security or functionality. |

| Impact Level | Count |
|---|---|
| Ad Hoc High | 1 |
| Automated High | 3 |
| Automated Medium | 6 |
| Automated Low | 23 |
| Automated Informational | 74 |

| Issue | Function Visibility |
|---|---|
| Type | function |
| Impact | Ad Hoc High |
| Confidence | High |
| Name | function _transferFrom( address sender, address recipient, uint256 amount ) public returns (bool) { |
| Source | TokenWithFlex.sol |
| Lines | 109-113 |
| Details | Function visibility is set to public that allows anyone to call the _transferFrom function bypassing approvals and move an arbitrary amount of tokens from any address. |

| Issue | arbitrary-send-eth |
|---|---|
| Type | node |
| Impact | High |
| Confidence | Medium |
| Name | distributor.deposit{value: amountETHReward}() |
| Source | TokenWithFlex.sol |
| Lines | 677-677 |

MasterKeyToken.swapBack() (TokenWithFlex.sol#644-682) sends eth to arbitrary user
Dangerous calls:
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)

function: swapBack
Source: TokenWithFlex.sol
Lines: 644-682

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

| Issue | reentrancy-eth |
|---|---|
| Type | node |
| Impact | High |
| Confidence | Medium |
| Name | _balances[address(deadWallet)] = _balances[address(deadWallet)].add(BFEE) |
| Source | TokenWithFlex.sol |
| Lines | 635-635 |

Reentrancy in MasterKeyToken._transferFrom(address,address,uint256)
(TokenWithFlex.sol#576-611):
External calls:
- swapBack() (TokenWithFlex.sol#589)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (TokenWithFlex.sol#900-906)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
External calls sending eth:
- swapBack() (TokenWithFlex.sol#589)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
- address(_marketingWalletAddress).transfer(amountETHMarketing) (TokenWithFlex.sol#671)
- address(_stakingWalletAddress).transfer(amountETHStaking) (TokenWithFlex.sol#674)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
State variables written after the call(s):
- _balances[sender] = _balances[sender].sub(amount) (TokenWithFlex.sol#592)
MasterKeyToken._balances (TokenWithFlex.sol#436) can be used in cross function reentrancies:
- MasterKeyToken._basicTransfer(address,address,uint256) (TokenWithFlex.sol#568-574)
- MasterKeyToken._transferFrom(address,address,uint256) (TokenWithFlex.sol#576-611)
- MasterKeyToken.balanceOf(address) (TokenWithFlex.sol#872-874)
- MasterKeyToken.constructor() (TokenWithFlex.sol#478-531)
- MasterKeyToken.getCirculatingSupply() (TokenWithFlex.sol#825-827)
- MasterKeyToken.getLiquidityBacking(uint256) (TokenWithFlex.sol#850-853)
- MasterKeyToken.takeFee(address,address,uint256) (TokenWithFlex.sol#613-642)
- MasterKeyToken.transfer(address,uint256) (TokenWithFlex.sol#552-557)
- _balances[recipient] = _balances[recipient].add(AmountReceived) (TokenWithFlex.sol#598)
MasterKeyToken._balances (TokenWithFlex.sol#436) can be used in cross function reentrancies:
- MasterKeyToken._basicTransfer(address,address,uint256) (TokenWithFlex.sol#568-574)
- MasterKeyToken._transferFrom(address,address,uint256) (TokenWithFlex.sol#576-611)
- MasterKeyToken.balanceOf(address) (TokenWithFlex.sol#872-874)
- MasterKeyToken.constructor() (TokenWithFlex.sol#478-531)
- MasterKeyToken.getCirculatingSupply() (TokenWithFlex.sol#825-827)
- MasterKeyToken.getLiquidityBacking(uint256) (TokenWithFlex.sol#850-853)
- MasterKeyToken.takeFee(address,address,uint256) (TokenWithFlex.sol#613-642)
- MasterKeyToken.transfer(address,uint256) (TokenWithFlex.sol#552-557)
- AmountReceived = takeFee(sender,recipient,amount) (TokenWithFlex.sol#594-596)
- _balances[address(this)] = _balances[address(this)].add(feeAmount) (TokenWithFlex.sol#630)
- _balances[address(deadWallet)] = _balances[address(deadWallet)].add(BFEE)
(TokenWithFlex.sol#635)
MasterKeyToken._balances (TokenWithFlex.sol#436) can be used in cross function reentrancies:
- MasterKeyToken._basicTransfer(address,address,uint256) (TokenWithFlex.sol#568-574)
- MasterKeyToken._transferFrom(address,address,uint256) (TokenWithFlex.sol#576-611)
- MasterKeyToken.balanceOf(address) (TokenWithFlex.sol#872-874)
- MasterKeyToken.constructor() (TokenWithFlex.sol#478-531)
- MasterKeyToken.getCirculatingSupply() (TokenWithFlex.sol#825-827)
- MasterKeyToken.getLiquidityBacking(uint256) (TokenWithFlex.sol#850-853)
- MasterKeyToken.takeFee(address,address,uint256) (TokenWithFlex.sol#613-642)
- MasterKeyToken.transfer(address,uint256) (TokenWithFlex.sol#552-557)

function: _transferFrom
Source: TokenWithFlex.sol
Lines: 576-611

node: swapBack()
Source: TokenWithFlex.sol
Lines: 589-589

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 900-906

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: swapBack()
Source: TokenWithFlex.sol
Lines: 589-589

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: TokenWithFlex.sol
Lines: 671-671

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: TokenWithFlex.sol
Lines: 674-674

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: _balances[sender] = _balances[sender].sub(amount)
Source: TokenWithFlex.sol
Lines: 592-592

node: _balances[recipient] = _balances[recipient].add(AmountReceived)
Source: TokenWithFlex.sol
Lines: 598-598

node: AmountReceived = takeFee(sender,recipient,amount)
Source: TokenWithFlex.sol

Lines: 594-596

node: _balances[address(this)] = _balances[address(this)].add(feeAmount)
Source: TokenWithFlex.sol
Lines: 630-630

node: _balances[address(deadWallet)] = _balances[address(deadWallet)].add(BFEE)
Source: TokenWithFlex.sol
Lines: 635-635

| Issue | unchecked-transfer |
|---|---|
| Type | node |
| Impact | High |
| Confidence | Medium |
| Name | BUSD.transfer(shareholder,amount) |
| Source | TokenWithFlex.sol |
| Lines | 281-281 |

DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286) ignores return value by BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)

function: distributeDividend
Source: TokenWithFlex.sol
Lines: 275-286

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

| Issue | reentrancy-no-eth |
|---|---|
| Type | node |
| Impact | Medium |
| Confidence | Medium |
| Name | shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount) |
| Source | TokenWithFlex.sol |
| Lines | 224-224 |

Reentrancy in DividendDistributor.setShare(address,uint256) (TokenWithFlex.sol#211-225):
External calls:
- distributeDividend(shareholder) (TokenWithFlex.sol#213)
- BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)
State variables written after the call(s):

- shares[shareholder].amount = amount (TokenWithFlex.sol#223)
DividendDistributor.shares (TokenWithFlex.sol#173) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286)
- DividendDistributor.getUnpaidEarnings(address) (TokenWithFlex.sol#292-301)
- DividendDistributor.setShare(address,uint256) (TokenWithFlex.sol#211-225)
- DividendDistributor.shares (TokenWithFlex.sol#173)
- shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
(TokenWithFlex.sol#224)
DividendDistributor.shares (TokenWithFlex.sol#173) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286)
- DividendDistributor.getUnpaidEarnings(address) (TokenWithFlex.sol#292-301)
- DividendDistributor.setShare(address,uint256) (TokenWithFlex.sol#211-225)
- DividendDistributor.shares (TokenWithFlex.sol#173)

function: setShare
Source: TokenWithFlex.sol
Lines: 211-225

node: distributeDividend(shareholder)
Source: TokenWithFlex.sol
Lines: 213-213

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: shares[shareholder].amount = amount
Source: TokenWithFlex.sol
Lines: 223-223

node: shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
Source: TokenWithFlex.sol
Lines: 224-224

| Issue | reentrancy-no-eth |
|---|---|
| Type | node |
| Impact | Medium |
| Confidence | Medium |
| Name | currentIndex ++ |
| Source | TokenWithFlex.sol |
| Lines | 266-266 |

Reentrancy in DividendDistributor.process(uint256) (TokenWithFlex.sol#246-269):
External calls:
- distributeDividend(shareholders[currentIndex]) (TokenWithFlex.sol#261)
- BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)
State variables written after the call(s):

- currentIndex = 0 (TokenWithFlex.sol#257)
DividendDistributor.currentIndex (TokenWithFlex.sol#179) can be used in cross function reentrancies:
- DividendDistributor.currentIndex (TokenWithFlex.sol#179)
- DividendDistributor.process(uint256) (TokenWithFlex.sol#246-269)
- currentIndex ++ (TokenWithFlex.sol#266)
DividendDistributor.currentIndex (TokenWithFlex.sol#179) can be used in cross function reentrancies:
- DividendDistributor.currentIndex (TokenWithFlex.sol#179)
- DividendDistributor.process(uint256) (TokenWithFlex.sol#246-269)

function: process
Source: TokenWithFlex.sol
Lines: 246-269

node: distributeDividend(shareholders[currentIndex])
Source: TokenWithFlex.sol
Lines: 261-261

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: currentIndex = 0
Source: TokenWithFlex.sol
Lines: 257-257

node: currentIndex ++
Source: TokenWithFlex.sol
Lines: 266-266

| Issue | reentrancy-no-eth |
|---|---|
| Type | node |
| Impact | Medium |
| Confidence | Medium |
| Name | shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount) |
| Source | TokenWithFlex.sol |
| Lines | 284-284 |

Reentrancy in DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286):
External calls:
- BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)
State variables written after the call(s):
- shares[shareholder].totalRealised = shares[shareholder].totalRealised.add(amount)
(TokenWithFlex.sol#283)
DividendDistributor.shares (TokenWithFlex.sol#173) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286)
- DividendDistributor.getUnpaidEarnings(address) (TokenWithFlex.sol#292-301)
- DividendDistributor.setShare(address,uint256) (TokenWithFlex.sol#211-225)

- DividendDistributor.shares (TokenWithFlex.sol#173)
- shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
(TokenWithFlex.sol#284)
DividendDistributor.shares (TokenWithFlex.sol#173) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286)
- DividendDistributor.getUnpaidEarnings(address) (TokenWithFlex.sol#292-301)
- DividendDistributor.setShare(address,uint256) (TokenWithFlex.sol#211-225)
- DividendDistributor.shares (TokenWithFlex.sol#173)

function: distributeDividend
Source: TokenWithFlex.sol
Lines: 275-286

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: shares[shareholder].totalRealised = shares[shareholder].totalRealised.add(amount)
Source: TokenWithFlex.sol
Lines: 283-283

node: shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
Source: TokenWithFlex.sol
Lines: 284-284

| Issue | uninitialized-local |
|---|---|
| Type | variable |
| Impact | Medium |
| Confidence | Medium |
| Name | feeAmount |
| Source | TokenWithFlex.sol |
| Lines | 615-615 |

MasterKeyToken.takeFee(address,address,uint256).feeAmount (TokenWithFlex.sol#615) is a local variable never initialized

variable: feeAmount
Source: TokenWithFlex.sol
Lines: 615-615

| Issue | uninitialized-local |
|---|---|
| Type | variable |
| Impact | Medium |

| | |
|---|---|
| Confidence | Medium |
| Name | BFEE |
| Source | TokenWithFlex.sol |
| Lines | 616-616 |

MasterKeyToken.takeFee(address,address,uint256).BFEE (TokenWithFlex.sol#616) is a local variable never initialized

variable: BFEE
Source: TokenWithFlex.sol
Lines: 616-616

| Issue | unused-return |
|---|---|
| Type | node |
| Impact | Medium |
| Confidence | Medium |
| Name | router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.ti |
| Source | TokenWithFlex.sol |
| Lines | 880-887 |

MasterKeyToken.addLiquidity(uint256,uint256) (TokenWithFlex.sol#876-889) ignores return value by router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp) (TokenWithFlex.sol#880-887)

function: addLiquidity
Source: TokenWithFlex.sol
Lines: 876-889

node: router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

| Issue | shadowing-local |
|---|---|
| Type | function |
| Impact | Low |
| Confidence | High |
| Name | owner |
| Source | TokenWithFlex.sol |
| Lines | 333-335 |

MasterKeyToken._spendAllowance(address,address,uint256).owner (TokenWithFlex.sol#720) shadows:
- Ownable.owner() (TokenWithFlex.sol#333-335) (function)

variable: owner
Source: TokenWithFlex.sol
Lines: 720-720

function: owner
Source: TokenWithFlex.sol
Lines: 333-335

| Issue | shadowing-local |
|---|---|
| Type | function |
| Impact | Low |
| Confidence | High |
| Name | owner |
| Source | TokenWithFlex.sol |
| Lines | 333-335 |

MasterKeyToken._approve(address,address,uint256).owner (TokenWithFlex.sol#735) shadows:
- Ownable.owner() (TokenWithFlex.sol#333-335) (function)

variable: owner
Source: TokenWithFlex.sol
Lines: 735-735

function: owner
Source: TokenWithFlex.sol
Lines: 333-335

| Issue | events-maths |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | totalSell = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking) |
| Source | TokenWithFlex.sol |
| Lines | 763-763 |

MasterKeyToken.setSellFee(uint256,uint256,uint256,uint256,uint256) (TokenWithFlex.sol#756-764) should emit an event for:
- sellLiquidityFee = _newLp (TokenWithFlex.sol#758)
- sellMarketingFee = _newMarketing (TokenWithFlex.sol#759)

- sellForBurn = _newburn (TokenWithFlex.sol#761)
- sellForstaking = _newStaking (TokenWithFlex.sol#762)
- totalSell = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking) (TokenWithFlex.sol#763)

function: setSellFee
Source: TokenWithFlex.sol
Lines: 756-764

node: sellLiquidityFee = _newLp
Source: TokenWithFlex.sol
Lines: 758-758

node: sellMarketingFee = _newMarketing
Source: TokenWithFlex.sol
Lines: 759-759

node: sellForBurn = _newburn
Source: TokenWithFlex.sol
Lines: 761-761

node: sellForstaking = _newStaking
Source: TokenWithFlex.sol
Lines: 762-762

node: totalSell = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
Source: TokenWithFlex.sol
Lines: 763-763

| Issue | events-maths |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | distributorGas = gas |
| Source | TokenWithFlex.sol |
| Lines | 822-822 |

MasterKeyToken.setDistributorSettings(uint256) (TokenWithFlex.sol#820-823) should emit an event for:
- distributorGas = gas (TokenWithFlex.sol#822)

function: setDistributorSettings
Source: TokenWithFlex.sol
Lines: 820-823

node: distributorGas = gas
Source: TokenWithFlex.sol

Lines: 822-822

| Issue | events-maths |
| --- | --- |
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | totalBuy = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking) |
| Source | TokenWithFlex.sol |
| Lines | 753-753 |

MasterKeyToken.setBuyFee(uint256,uint256,uint256,uint256,uint256) (TokenWithFlex.sol#747-754) should emit an event for:
- buyLiquidityFee = _newLp (TokenWithFlex.sol#748)
- buyMarketingFee = _newMarketing (TokenWithFlex.sol#749)
- buyForBurn = _newburn (TokenWithFlex.sol#751)
- buyForstaking = _newStaking (TokenWithFlex.sol#752)
- totalBuy = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking) (TokenWithFlex.sol#753)

function: setBuyFee
Source: TokenWithFlex.sol
Lines: 747-754

node: buyLiquidityFee = _newLp
Source: TokenWithFlex.sol
Lines: 748-748

node: buyMarketingFee = _newMarketing
Source: TokenWithFlex.sol
Lines: 749-749

node: buyForBurn = _newburn
Source: TokenWithFlex.sol
Lines: 751-751

node: buyForstaking = _newStaking
Source: TokenWithFlex.sol
Lines: 752-752

node: totalBuy = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
Source: TokenWithFlex.sol
Lines: 753-753

| Issue | events-maths |
| --- | --- |

| | |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | swapTokensAtAmount = _value |
| Source | TokenWithFlex.sol |
| Lines | 865-865 |

MasterKeyToken.setMinSwapAmount(uint256) (TokenWithFlex.sol#864-866) should emit an event for:
- swapTokensAtAmount = _value (TokenWithFlex.sol#865)

function: setMinSwapAmount
Source: TokenWithFlex.sol
Lines: 864-866

node: swapTokensAtAmount = _value
Source: TokenWithFlex.sol
Lines: 865-865

| **Issue** | **missing-zero-check** |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | _marketingWalletAddress = _marketing |
| Source | TokenWithFlex.sol |
| Lines | 809-809 |

MasterKeyToken.setMarketingWallet(address)._marketing (TokenWithFlex.sol#808) lacks a zero-check on :
- _marketingWalletAddress = _marketing (TokenWithFlex.sol#809)

variable: _marketing
Source: TokenWithFlex.sol
Lines: 808-808

node: _marketingWalletAddress = _marketing
Source: TokenWithFlex.sol
Lines: 809-809

| **Issue** | **missing-zero-check** |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |

| | |
|---|---|
| Name | _stakingWalletAddress = _staking |
| Source | TokenWithFlex.sol |
| Lines | 813-813 |

MasterKeyToken.setStakingWallet(address)._staking (TokenWithFlex.sol#812) lacks a zero-check on :
- _stakingWalletAddress = _staking (TokenWithFlex.sol#813)

variable: _staking
Source: TokenWithFlex.sol
Lines: 812-812

node: _stakingWalletAddress = _staking
Source: TokenWithFlex.sol
Lines: 813-813

| Issue | missing-zero-check |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | BUSDDividendReceiver = _address |
| Source | TokenWithFlex.sol |
| Lines | 767-767 |

MasterKeyToken.setDistributor(address)._address (TokenWithFlex.sol#765) lacks a zero-check on :
- BUSDDividendReceiver = _address (TokenWithFlex.sol#767)

variable: _address
Source: TokenWithFlex.sol
Lines: 765-765

node: BUSDDividendReceiver = _address
Source: TokenWithFlex.sol
Lines: 767-767

| Issue | missing-zero-check |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | BUSDMarketDistributor = _newMarketDividend |
| Source | TokenWithFlex.sol |
| Lines | 772-772 |

MasterKeyToken.setMarketDividend(address)._newMarketDividend (TokenWithFlex.sol#770) lacks a zero-check on :
- BUSDMarketDistributor = _newMarketDividend (TokenWithFlex.sol#772)

variable: _newMarketDividend
Source: TokenWithFlex.sol
Lines: 770-770

node: BUSDMarketDistributor = _newMarketDividend
Source: TokenWithFlex.sol
Lines: 772-772

| Issue | missing-zero-check |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | _liquidityReciever = _liquidity |
| Source | TokenWithFlex.sol |
| Lines | 817-817 |

MasterKeyToken.setLiquidityWallet(address)._liquidity (TokenWithFlex.sol#816) lacks a zero-check on :
- _liquidityReciever = _liquidity (TokenWithFlex.sol#817)

variable: _liquidity
Source: TokenWithFlex.sol
Lines: 816-816

node: _liquidityReciever = _liquidity
Source: TokenWithFlex.sol
Lines: 817-817

| Issue | missing-zero-check |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | deadWallet = _address |
| Source | TokenWithFlex.sol |
| Lines | 705-705 |

MasterKeyToken.setDeadWallet(address)._address (TokenWithFlex.sol#704) lacks a zero-check on :
- deadWallet = _address (TokenWithFlex.sol#705)

variable: _address
Source: TokenWithFlex.sol
Lines: 704-704

node: deadWallet = _address
Source: TokenWithFlex.sol
Lines: 705-705

| Issue | missing-zero-check |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | address(_receiver).transfer(balance) |
| Source | TokenWithFlex.sol |
| Lines | 797-797 |

MasterKeyToken.clearStuckBalance(address)._receiver (TokenWithFlex.sol#795) lacks a zero-check on :
- address(_receiver).transfer(balance) (TokenWithFlex.sol#797)

variable: _receiver
Source: TokenWithFlex.sol
Lines: 795-795

node: address(_receiver).transfer(balance)
Source: TokenWithFlex.sol
Lines: 797-797

| Issue | missing-zero-check |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | pair = _address |
| Source | TokenWithFlex.sol |
| Lines | 839-839 |

MasterKeyToken.setLP(address)._address (TokenWithFlex.sol#837) lacks a zero-check on :
- pair = _address (TokenWithFlex.sol#839)

variable: _address
Source: TokenWithFlex.sol
Lines: 837-837

node: pair = _address
Source: TokenWithFlex.sol
Lines: 839-839

| Issue | calls-loop |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | BUSD.transfer(shareholder,amount) |
| Source | TokenWithFlex.sol |
| Lines | 281-281 |

DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286) has external calls inside a loop: BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)

function: distributeDividend
Source: TokenWithFlex.sol
Lines: 275-286

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

| Issue | reentrancy-benign |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | _allowances[owner][spender] = amount |
| Source | TokenWithFlex.sol |
| Lines | 739-739 |

Reentrancy in MasterKeyToken.swapBack() (TokenWithFlex.sol#644-682):
External calls:
- swapTokensForEth(tokensForSwap) (TokenWithFlex.sol#660)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (TokenWithFlex.sol#900-906)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
- addLiquidity(tokensForLP,amountETHLiquidity) (TokenWithFlex.sol#680)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
External calls sending eth:
- address(_marketingWalletAddress).transfer(amountETHMarketing) (TokenWithFlex.sol#671)

- address(_stakingWalletAddress).transfer(amountETHStaking) (TokenWithFlex.sol#674)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
- addLiquidity(tokensForLP,amountETHLiquidity) (TokenWithFlex.sol#680)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
State variables written after the call(s):
- addLiquidity(tokensForLP,amountETHLiquidity) (TokenWithFlex.sol#680)
- _allowances[owner][spender] = amount (TokenWithFlex.sol#739)

function: swapBack
Source: TokenWithFlex.sol
Lines: 644-682

node: swapTokensForEth(tokensForSwap)
Source: TokenWithFlex.sol
Lines: 660-660

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 900-906

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: TokenWithFlex.sol
Lines: 680-680

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: swapTokensForEth(tokensForSwap)
Source: TokenWithFlex.sol
Lines: 660-660

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 900-906

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: TokenWithFlex.sol
Lines: 680-680

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: TokenWithFlex.sol
Lines: 680-680

node: _allowances[owner][spender] = amount
Source: TokenWithFlex.sol
Lines: 739-739

| Issue | reentrancy-benign |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | totalDividends = totalDividends.add(amount) |
| Source | TokenWithFlex.sol |
| Lines | 242-242 |

Reentrancy in DividendDistributor.deposit() (TokenWithFlex.sol#231-244):
External calls:
- router.swapExactETHForTokensSupportingFeeOnTransferTokens{value:
msg.value}(0,path,address(this),block.timestamp) (TokenWithFlex.sol#238)
State variables written after the call(s):
- dividendsPerShare =
dividendsPerShare.add(dividendsPerShareAccuracyFactor.mul(amount).div(totalShares))
(TokenWithFlex.sol#243)
- totalDividends = totalDividends.add(amount) (TokenWithFlex.sol#242)

function: deposit
Source: TokenWithFlex.sol
Lines: 231-244

node: router.swapExactETHForTokensSupportingFeeOnTransferTokens{value:
msg.value}(0,path,address(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 238-238

node: dividendsPerShare =
dividendsPerShare.add(dividendsPerShareAccuracyFactor.mul(amount).div(totalShares))
Source: TokenWithFlex.sol
Lines: 243-243

node: totalDividends = totalDividends.add(amount)
Source: TokenWithFlex.sol

Lines: 242-242

| Issue | reentrancy-benign |
| --- | --- |
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | totalShares = totalShares.sub(shares[shareholder].amount).add(amount) |
| Source | TokenWithFlex.sol |
| Lines | 222-222 |

Reentrancy in DividendDistributor.setShare(address,uint256) (TokenWithFlex.sol#211-225):
External calls:
- distributeDividend(shareholder) (TokenWithFlex.sol#213)
- BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)
State variables written after the call(s):
- addShareholder(shareholder) (TokenWithFlex.sol#217)
- shareholderIndexes[shareholder] = shareholders.length (TokenWithFlex.sol#308)
- removeShareholder(shareholder) (TokenWithFlex.sol#219)
- shareholderIndexes[shareholders[shareholders.length - 1]] = shareholderIndexes[shareholder] (TokenWithFlex.sol#314)
- addShareholder(shareholder) (TokenWithFlex.sol#217)
- shareholders.push(shareholder) (TokenWithFlex.sol#309)
- removeShareholder(shareholder) (TokenWithFlex.sol#219)
- shareholders[shareholderIndexes[shareholder]] = shareholders[shareholders.length - 1] (TokenWithFlex.sol#313)
- shareholders.pop() (TokenWithFlex.sol#315)
- totalShares = totalShares.sub(shares[shareholder].amount).add(amount) (TokenWithFlex.sol#222)

function: setShare
Source: TokenWithFlex.sol
Lines: 211-225

node: distributeDividend(shareholder)
Source: TokenWithFlex.sol
Lines: 213-213

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: distributeDividend(shareholder)
Source: TokenWithFlex.sol
Lines: 213-213

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: addShareholder(shareholder)
Source: TokenWithFlex.sol
Lines: 217-217

node: shareholderIndexes[shareholder] = shareholders.length
Source: TokenWithFlex.sol
Lines: 308-308

node: removeShareholder(shareholder)
Source: TokenWithFlex.sol
Lines: 219-219

node: shareholderIndexes[shareholders[shareholders.length - 1]] = shareholderIndexes[shareholder]
Source: TokenWithFlex.sol
Lines: 314-314

node: addShareholder(shareholder)
Source: TokenWithFlex.sol
Lines: 217-217

node: shareholders.push(shareholder)
Source: TokenWithFlex.sol
Lines: 309-309

node: removeShareholder(shareholder)
Source: TokenWithFlex.sol
Lines: 219-219

node: shareholders[shareholderIndexes[shareholder]] = shareholders[shareholders.length - 1]
Source: TokenWithFlex.sol
Lines: 313-313

node: shareholders.pop()
Source: TokenWithFlex.sol
Lines: 315-315

node: totalShares = totalShares.sub(shares[shareholder].amount).add(amount)
Source: TokenWithFlex.sol
Lines: 222-222

| Issue | reentrancy-benign |
| --- | --- |
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | shareholderClaims[shareholder] = block.timestamp |
| Source | TokenWithFlex.sol |

| | |
|---|---|
| Lines | 282-282 |

Reentrancy in DividendDistributor.distributeDividend(address) (TokenWithFlex.sol#275-286):
External calls:
- BUSD.transfer(shareholder,amount) (TokenWithFlex.sol#281)
State variables written after the call(s):
- shareholderClaims[shareholder] = block.timestamp (TokenWithFlex.sol#282)

function: distributeDividend
Source: TokenWithFlex.sol
Lines: 275-286

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: BUSD.transfer(shareholder,amount)
Source: TokenWithFlex.sol
Lines: 281-281

node: shareholderClaims[shareholder] = block.timestamp
Source: TokenWithFlex.sol
Lines: 282-282

| **Issue** | **reentrancy-events** |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | Transfer(sender,recipient,AmountReceived) |
| Source | TokenWithFlex.sol |
| Lines | 609-609 |

Reentrancy in MasterKeyToken._transferFrom(address,address,uint256)
(TokenWithFlex.sol#576-611):
External calls:
- swapBack() (TokenWithFlex.sol#589)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (TokenWithFlex.sol#900-906)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
- distributor.setShare(sender,balanceOf(sender)) (TokenWithFlex.sol#600)
- distributor.setShare(recipient,balanceOf(recipient)) (TokenWithFlex.sol#601)
- marketDistributor.setShare(sender,balanceOf(sender)) (TokenWithFlex.sol#604)
- marketDistributor.setShare(recipient,balanceOf(recipient)) (TokenWithFlex.sol#605)
- distributor.process(distributorGas) (TokenWithFlex.sol#607)
External calls sending eth:

- swapBack() (TokenWithFlex.sol#589)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
- address(_marketingWalletAddress).transfer(amountETHMarketing) (TokenWithFlex.sol#671)
- address(_stakingWalletAddress).transfer(amountETHStaking) (TokenWithFlex.sol#674)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
Event emitted after the call(s):
- Transfer(sender,recipient,AmountReceived) (TokenWithFlex.sol#609)

function: _transferFrom
Source: TokenWithFlex.sol
Lines: 576-611

node: swapBack()
Source: TokenWithFlex.sol
Lines: 589-589

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 900-906

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: distributor.setShare(sender,balanceOf(sender))
Source: TokenWithFlex.sol
Lines: 600-600

node: distributor.setShare(recipient,balanceOf(recipient))
Source: TokenWithFlex.sol
Lines: 601-601

node: marketDistributor.setShare(sender,balanceOf(sender))
Source: TokenWithFlex.sol
Lines: 604-604

node: marketDistributor.setShare(recipient,balanceOf(recipient))
Source: TokenWithFlex.sol
Lines: 605-605

node: distributor.process(distributorGas)
Source: TokenWithFlex.sol
Lines: 607-607

node: swapBack()
Source: TokenWithFlex.sol
Lines: 589-589

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: TokenWithFlex.sol
Lines: 671-671

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: TokenWithFlex.sol
Lines: 674-674

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: Transfer(sender,recipient,AmountReceived)
Source: TokenWithFlex.sol
Lines: 609-609

| Issue | reentrancy-events |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | addLiquidity(tokensForLP,amountETHLiquidity) |
| Source | TokenWithFlex.sol |
| Lines | 680-680 |

Reentrancy in MasterKeyToken.swapBack() (TokenWithFlex.sol#644-682):
External calls:
- swapTokensForEth(tokensForSwap) (TokenWithFlex.sol#660)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (TokenWithFlex.sol#900-906)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
- addLiquidity(tokensForLP,amountETHLiquidity) (TokenWithFlex.sol#680)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
External calls sending eth:
- address(_marketingWalletAddress).transfer(amountETHMarketing) (TokenWithFlex.sol#671)
- address(_stakingWalletAddress).transfer(amountETHStaking) (TokenWithFlex.sol#674)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
- addLiquidity(tokensForLP,amountETHLiquidity) (TokenWithFlex.sol#680)

- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
Event emitted after the call(s):
- Approval(owner,spender,amount) (TokenWithFlex.sol#740)
- addLiquidity(tokensForLP,amountETHLiquidity) (TokenWithFlex.sol#680)

function: swapBack
Source: TokenWithFlex.sol
Lines: 644-682

node: swapTokensForEth(tokensForSwap)
Source: TokenWithFlex.sol
Lines: 660-660

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 900-906

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: TokenWithFlex.sol
Lines: 680-680

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: TokenWithFlex.sol
Lines: 671-671

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: TokenWithFlex.sol
Lines: 674-674

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: TokenWithFlex.sol
Lines: 680-680

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol

Lines: 880-887

node: Approval(owner,spender,amount)
Source: TokenWithFlex.sol
Lines: 740-740

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: TokenWithFlex.sol
Lines: 680-680

| Issue | reentrancy-events |
|---|---|
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | AmountReceived = takeFee(sender,recipient,amount) |
| Source | TokenWithFlex.sol |
| Lines | 594-596 |

Reentrancy in MasterKeyToken._transferFrom(address,address,uint256)
(TokenWithFlex.sol#576-611):
External calls:
- swapBack() (TokenWithFlex.sol#589)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (TokenWithFlex.sol#900-906)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
External calls sending eth:
- swapBack() (TokenWithFlex.sol#589)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(TokenWithFlex.sol#880-887)
- address(_marketingWalletAddress).transfer(amountETHMarketing) (TokenWithFlex.sol#671)
- address(_stakingWalletAddress).transfer(amountETHStaking) (TokenWithFlex.sol#674)
- distributor.deposit{value: amountETHReward}() (TokenWithFlex.sol#677)
Event emitted after the call(s):
- Transfer(sender,address(this),feeAmount) (TokenWithFlex.sol#631)
- AmountReceived = takeFee(sender,recipient,amount) (TokenWithFlex.sol#594-596)
- Transfer(sender,address(deadWallet),BFEE) (TokenWithFlex.sol#636)
- AmountReceived = takeFee(sender,recipient,amount) (TokenWithFlex.sol#594-596)

function: _transferFrom
Source: TokenWithFlex.sol
Lines: 576-611

node: swapBack()
Source: TokenWithFlex.sol
Lines: 589-589

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: TokenWithFlex.sol
Lines: 900-906

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: swapBack()
Source: TokenWithFlex.sol
Lines: 589-589

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: TokenWithFlex.sol
Lines: 880-887

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: TokenWithFlex.sol
Lines: 671-671

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: TokenWithFlex.sol
Lines: 674-674

node: distributor.deposit{value: amountETHReward}()
Source: TokenWithFlex.sol
Lines: 677-677

node: Transfer(sender,address(this),feeAmount)
Source: TokenWithFlex.sol
Lines: 631-631

node: AmountReceived = takeFee(sender,recipient,amount)
Source: TokenWithFlex.sol
Lines: 594-596

node: Transfer(sender,address(deadWallet),BFEE)
Source: TokenWithFlex.sol
Lines: 636-636

node: AmountReceived = takeFee(sender,recipient,amount)
Source: TokenWithFlex.sol

Lines: 594-596

| Issue | timestamp |
| --- | --- |
| Type | node |
| Impact | Low |
| Confidence | Medium |
| Name | shareholderClaims[shareholder] + minPeriod < block.timestamp && getUnpaidEarnings(shareholder) |
| Source | TokenWithFlex.sol |
| Lines | 272-272 |

DividendDistributor.shouldDistribute(address) (TokenWithFlex.sol#271-273) uses timestamp for comparisons
Dangerous comparisons:
- shareholderClaims[shareholder] + minPeriod < block.timestamp && getUnpaidEarnings(shareholder) > minDistribution (TokenWithFlex.sol#272)

function: shouldDistribute
Source: TokenWithFlex.sol
Lines: 271-273

node: shareholderClaims[shareholder] + minPeriod < block.timestamp && getUnpaidEarnings(shareholder) > minDistribution
Source: TokenWithFlex.sol
Lines: 272-272