ZEROSEC AUDIT REPORT

PROJECT NAME: MASTER KEY FINANCE

CONTRACT: MKFTOKEN.SOL

WEBSITE: HTTPS://MASTERKEY.FINANCE

INTRODUCTION

Auditing Firm	Zero Sec LLC
Client Firm	Master Key Finance
Methodology	Manual and Automated Analysis
Language	Solidity
Contract	No Deployment Address
Blockchain	Binance Smart Chain
Ownership	Centralized Ownership
Website	https://masterkey.finance
Telegram	https://t.me/MasterKeyOfficial
Twitter	https://www.x.com/MasterKey_Fin
Instagram	https://instagram.com/mkf__official
Report Date	September 9, 2024

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	0
Automated High	3
Automated Medium	6
Automated Low	23
Automated Informational	74

Issue	arbitrary-send-eth
Type	node
Impact	High
Confidence	Medium
Name	distributor.deposit{value: amountETHReward}()
Source	MKFToken.sol
Lines	681-681

MasterKeyToken.swapBack() (MKFToken.sol#648-686) sends eth to arbitrary user
Dangerous calls:
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)

function: swapBack
Source: MKFToken.sol
Lines: 648-686

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

Issue	reentrancy-eth
Type	node
Impact	High
Confidence	Medium
Name	_balances[address(deadWallet)] = _balances[address(deadWallet)].add(BFEE)
Source	MKFToken.sol
Lines	639-639

Reentrancy in MasterKeyToken._transferFrom(address,address,uint256) (MKFToken.sol#580-615):
External calls:
- swapBack() (MKFToken.sol#593)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (MKFToken.sol#904-910)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
External calls sending eth:
- swapBack() (MKFToken.sol#593)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
- address(_marketingWalletAddress).transfer(amountETHMarketing) (MKFToken.sol#675)

- address(_stakingWalletAddress).transfer(amountETHStaking) (MKFToken.sol#678)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
State variables written after the call(s):
- _balances[sender] = _balances[sender].sub(amount) (MKFToken.sol#596)
MasterKeyToken._balances (MKFToken.sol#440) can be used in cross function reentrancies:
- MasterKeyToken._basicTransfer(address,address,uint256) (MKFToken.sol#572-578)
- MasterKeyToken._transferFrom(address,address,uint256) (MKFToken.sol#580-615)
- MasterKeyToken.balanceOf(address) (MKFToken.sol#876-878)
- MasterKeyToken.constructor() (MKFToken.sol#482-535)
- MasterKeyToken.getCirculatingSupply() (MKFToken.sol#829-831)
- MasterKeyToken.getLiquidityBacking(uint256) (MKFToken.sol#854-857)
- MasterKeyToken.takeFee(address,address,uint256) (MKFToken.sol#617-646)
- MasterKeyToken.transfer(address,uint256) (MKFToken.sol#556-561)
- _balances[recipient] = _balances[recipient].add(AmountReceived) (MKFToken.sol#602)
MasterKeyToken._balances (MKFToken.sol#440) can be used in cross function reentrancies:
- MasterKeyToken._basicTransfer(address,address,uint256) (MKFToken.sol#572-578)
- MasterKeyToken._transferFrom(address,address,uint256) (MKFToken.sol#580-615)
- MasterKeyToken.balanceOf(address) (MKFToken.sol#876-878)
- MasterKeyToken.constructor() (MKFToken.sol#482-535)
- MasterKeyToken.getCirculatingSupply() (MKFToken.sol#829-831)
- MasterKeyToken.getLiquidityBacking(uint256) (MKFToken.sol#854-857)
- MasterKeyToken.takeFee(address,address,uint256) (MKFToken.sol#617-646)
- MasterKeyToken.transfer(address,uint256) (MKFToken.sol#556-561)
- AmountReceived = takeFee(sender,recipient,amount) (MKFToken.sol#598-600)
- _balances[address(this)] = _balances[address(this)].add(feeAmount) (MKFToken.sol#634)
- _balances[address(deadWallet)] = _balances[address(deadWallet)].add(BFEE) (MKFToken.sol#639)
MasterKeyToken._balances (MKFToken.sol#440) can be used in cross function reentrancies:
- MasterKeyToken._basicTransfer(address,address,uint256) (MKFToken.sol#572-578)
- MasterKeyToken._transferFrom(address,address,uint256) (MKFToken.sol#580-615)
- MasterKeyToken.balanceOf(address) (MKFToken.sol#876-878)
- MasterKeyToken.constructor() (MKFToken.sol#482-535)
- MasterKeyToken.getCirculatingSupply() (MKFToken.sol#829-831)
- MasterKeyToken.getLiquidityBacking(uint256) (MKFToken.sol#854-857)
- MasterKeyToken.takeFee(address,address,uint256) (MKFToken.sol#617-646)
- MasterKeyToken.transfer(address,uint256) (MKFToken.sol#556-561)

function: _transferFrom
Source: MKFToken.sol
Lines: 580-615

node: swapBack()
Source: MKFToken.sol
Lines: 593-593

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: MKFToken.sol

Lines: 904-910

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: swapBack()
Source: MKFToken.sol
Lines: 593-593

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: MKFToken.sol
Lines: 675-675

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: MKFToken.sol
Lines: 678-678

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: _balances[sender] = _balances[sender].sub(amount)
Source: MKFToken.sol
Lines: 596-596

node: _balances[recipient] = _balances[recipient].add(AmountReceived)
Source: MKFToken.sol
Lines: 602-602

node: AmountReceived = takeFee(sender,recipient,amount)
Source: MKFToken.sol
Lines: 598-600

node: _balances[address(this)] = _balances[address(this)].add(feeAmount)
Source: MKFToken.sol
Lines: 634-634

node: _balances[address(deadWallet)] = _balances[address(deadWallet)].add(BFEE)
Source: MKFToken.sol
Lines: 639-639

Issue	unchecked-transfer
Type	node

Impact	High
Confidence	Medium
Name	BUSD.transfer(shareholder,amount)
Source	MKFToken.sol
Lines	285-285

DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290) ignores return value by BUSD.transfer(shareholder,amount) (MKFToken.sol#285)

function: distributeDividend
Source: MKFToken.sol
Lines: 279-290

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

Issue	**reentrancy-no-eth**
Type	node
Impact	Medium
Confidence	Medium
Name	shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
Source	MKFToken.sol
Lines	288-288

Reentrancy in DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290):
External calls:
- BUSD.transfer(shareholder,amount) (MKFToken.sol#285)
State variables written after the call(s):
- shares[shareholder].totalRealised = shares[shareholder].totalRealised.add(amount) (MKFToken.sol#287)
DividendDistributor.shares (MKFToken.sol#177) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290)
- DividendDistributor.getUnpaidEarnings(address) (MKFToken.sol#296-305)
- DividendDistributor.setShare(address,uint256) (MKFToken.sol#215-229)
- DividendDistributor.shares (MKFToken.sol#177)
- shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount) (MKFToken.sol#288)
DividendDistributor.shares (MKFToken.sol#177) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290)
- DividendDistributor.getUnpaidEarnings(address) (MKFToken.sol#296-305)
- DividendDistributor.setShare(address,uint256) (MKFToken.sol#215-229)
- DividendDistributor.shares (MKFToken.sol#177)

function: distributeDividend
Source: MKFToken.sol

Lines: 279-290

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: shares[shareholder].totalRealised = shares[shareholder].totalRealised.add(amount)
Source: MKFToken.sol
Lines: 287-287

node: shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
Source: MKFToken.sol
Lines: 288-288

Issue	reentrancy-no-eth
Type	node
Impact	Medium
Confidence	Medium
Name	shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
Source	MKFToken.sol
Lines	228-228

Reentrancy in DividendDistributor.setShare(address,uint256) (MKFToken.sol#215-229):
External calls:
- distributeDividend(shareholder) (MKFToken.sol#217)
- BUSD.transfer(shareholder,amount) (MKFToken.sol#285)
State variables written after the call(s):
- shares[shareholder].amount = amount (MKFToken.sol#227)
DividendDistributor.shares (MKFToken.sol#177) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290)
- DividendDistributor.getUnpaidEarnings(address) (MKFToken.sol#296-305)
- DividendDistributor.setShare(address,uint256) (MKFToken.sol#215-229)
- DividendDistributor.shares (MKFToken.sol#177)
- shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
(MKFToken.sol#228)
DividendDistributor.shares (MKFToken.sol#177) can be used in cross function reentrancies:
- DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290)
- DividendDistributor.getUnpaidEarnings(address) (MKFToken.sol#296-305)
- DividendDistributor.setShare(address,uint256) (MKFToken.sol#215-229)
- DividendDistributor.shares (MKFToken.sol#177)

function: setShare
Source: MKFToken.sol
Lines: 215-229

node: distributeDividend(shareholder)
Source: MKFToken.sol

Lines: 217-217

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: shares[shareholder].amount = amount
Source: MKFToken.sol
Lines: 227-227

node: shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount)
Source: MKFToken.sol
Lines: 228-228

Issue	reentrancy-no-eth
Type	node
Impact	Medium
Confidence	Medium
Name	currentIndex ++
Source	MKFToken.sol
Lines	270-270

Reentrancy in DividendDistributor.process(uint256) (MKFToken.sol#250-273):
External calls:
- distributeDividend(shareholders[currentIndex]) (MKFToken.sol#265)
- BUSD.transfer(shareholder,amount) (MKFToken.sol#285)
State variables written after the call(s):
- currentIndex = 0 (MKFToken.sol#261)
DividendDistributor.currentIndex (MKFToken.sol#183) can be used in cross function reentrancies:
- DividendDistributor.currentIndex (MKFToken.sol#183)
- DividendDistributor.process(uint256) (MKFToken.sol#250-273)
- currentIndex ++ (MKFToken.sol#270)
DividendDistributor.currentIndex (MKFToken.sol#183) can be used in cross function reentrancies:
- DividendDistributor.currentIndex (MKFToken.sol#183)
- DividendDistributor.process(uint256) (MKFToken.sol#250-273)

function: process
Source: MKFToken.sol
Lines: 250-273

node: distributeDividend(shareholders[currentIndex])
Source: MKFToken.sol
Lines: 265-265

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: currentIndex = 0
Source: MKFToken.sol
Lines: 261-261

node: currentIndex ++
Source: MKFToken.sol
Lines: 270-270

Issue	uninitialized-local
Type	variable
Impact	Medium
Confidence	Medium
Name	feeAmount
Source	MKFToken.sol
Lines	619-619

MasterKeyToken.takeFee(address,address,uint256).feeAmount (MKFToken.sol#619) is a local variable never initialized

variable: feeAmount
Source: MKFToken.sol
Lines: 619-619

Issue	uninitialized-local
Type	variable
Impact	Medium
Confidence	Medium
Name	BFEE
Source	MKFToken.sol
Lines	620-620

MasterKeyToken.takeFee(address,address,uint256).BFEE (MKFToken.sol#620) is a local variable never initialized

variable: BFEE
Source: MKFToken.sol
Lines: 620-620

Issue	unused-return

Type	node
Impact	Medium
Confidence	Medium
Name	router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.ti
Source	MKFToken.sol
Lines	884-891

MasterKeyToken.addLiquidity(uint256,uint256) (MKFToken.sol#880-893) ignores return value by router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp) (MKFToken.sol#884-891)

function: addLiquidity
Source: MKFToken.sol
Lines: 880-893

node: router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

Issue	shadowing-local
Type	function
Impact	Low
Confidence	High
Name	owner
Source	MKFToken.sol
Lines	337-339

MasterKeyToken._spendAllowance(address,address,uint256).owner (MKFToken.sol#724) shadows:
- Ownable.owner() (MKFToken.sol#337-339) (function)

variable: owner
Source: MKFToken.sol
Lines: 724-724

function: owner
Source: MKFToken.sol
Lines: 337-339

Issue	shadowing-local
Type	function

Impact	Low
Confidence	High
Name	owner
Source	MKFToken.sol
Lines	337-339

MasterKeyToken._approve(address,address,uint256).owner (MKFToken.sol#739) shadows:
- Ownable.owner() (MKFToken.sol#337-339) (function)

variable: owner
Source: MKFToken.sol
Lines: 739-739

function: owner
Source: MKFToken.sol
Lines: 337-339

Issue	events-maths
Type	node
Impact	Low
Confidence	Medium
Name	distributorGas = gas
Source	MKFToken.sol
Lines	826-826

MasterKeyToken.setDistributorSettings(uint256) (MKFToken.sol#824-827) should emit an event for:
- distributorGas = gas (MKFToken.sol#826)

function: setDistributorSettings
Source: MKFToken.sol
Lines: 824-827

node: distributorGas = gas
Source: MKFToken.sol
Lines: 826-826

Issue	events-maths
Type	node
Impact	Low
Confidence	Medium
Name	swapTokensAtAmount = _value

Source	MKFToken.sol
Lines	869-869

MasterKeyToken.setMinSwapAmount(uint256) (MKFToken.sol#868-870) should emit an event for:
- swapTokensAtAmount = _value (MKFToken.sol#869)

function: setMinSwapAmount
Source: MKFToken.sol
Lines: 868-870

node: swapTokensAtAmount = _value
Source: MKFToken.sol
Lines: 869-869

Issue	events-maths
Type	node
Impact	Low
Confidence	Medium
Name	totalSell = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
Source	MKFToken.sol
Lines	767-767

MasterKeyToken.setSellFee(uint256,uint256,uint256,uint256,uint256) (MKFToken.sol#760-768) should emit an event for:
- sellLiquidityFee = _newLp (MKFToken.sol#762)
- sellMarketingFee = _newMarketing (MKFToken.sol#763)
- sellForBurn = _newburn (MKFToken.sol#765)
- sellForstaking = _newStaking (MKFToken.sol#766)
- totalSell = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking) (MKFToken.sol#767)

function: setSellFee
Source: MKFToken.sol
Lines: 760-768

node: sellLiquidityFee = _newLp
Source: MKFToken.sol
Lines: 762-762

node: sellMarketingFee = _newMarketing
Source: MKFToken.sol
Lines: 763-763

node: sellForBurn = _newburn
Source: MKFToken.sol
Lines: 765-765

node: sellForstaking = _newStaking
Source: MKFToken.sol
Lines: 766-766

node: totalSell = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
Source: MKFToken.sol
Lines: 767-767

Issue	events-maths
Type	node
Impact	Low
Confidence	Medium
Name	totalBuy = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
Source	MKFToken.sol
Lines	757-757

MasterKeyToken.setBuyFee(uint256,uint256,uint256,uint256,uint256) (MKFToken.sol#751-758)
should emit an event for:
- buyLiquidityFee = _newLp (MKFToken.sol#752)
- buyMarketingFee = _newMarketing (MKFToken.sol#753)
- buyForBurn = _newburn (MKFToken.sol#755)
- buyForstaking = _newStaking (MKFToken.sol#756)
- totalBuy = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
(MKFToken.sol#757)

function: setBuyFee
Source: MKFToken.sol
Lines: 751-758

node: buyLiquidityFee = _newLp
Source: MKFToken.sol
Lines: 752-752

node: buyMarketingFee = _newMarketing
Source: MKFToken.sol
Lines: 753-753

node: buyForBurn = _newburn
Source: MKFToken.sol
Lines: 755-755

node: buyForstaking = _newStaking
Source: MKFToken.sol
Lines: 756-756

node: totalBuy = _newLp.add(_newMarketing).add(_newReward).add(_newburn).add(_newStaking)
Source: MKFToken.sol
Lines: 757-757

Issue	missing-zero-check
Type	node
Impact	Low
Confidence	Medium
Name	_stakingWalletAddress = _staking
Source	MKFToken.sol
Lines	817-817

MasterKeyToken.setStakingWallet(address)._staking (MKFToken.sol#816) lacks a zero-check on :
- _stakingWalletAddress = _staking (MKFToken.sol#817)

variable: _staking
Source: MKFToken.sol
Lines: 816-816

node: _stakingWalletAddress = _staking
Source: MKFToken.sol
Lines: 817-817

Issue	missing-zero-check
Type	node
Impact	Low
Confidence	Medium
Name	BUSDMarketDistributor = _newMarketDividend
Source	MKFToken.sol
Lines	776-776

MasterKeyToken.setMarketDividend(address)._newMarketDividend (MKFToken.sol#774) lacks a zero-check on :
- BUSDMarketDistributor = _newMarketDividend (MKFToken.sol#776)

variable: _newMarketDividend
Source: MKFToken.sol
Lines: 774-774

node: BUSDMarketDistributor = _newMarketDividend
Source: MKFToken.sol
Lines: 776-776

Issue	missing-zero-check

Type	node
Impact	Low
Confidence	Medium
Name	_liquidityReciever = _liquidity
Source	MKFToken.sol
Lines	821-821

MasterKeyToken.setLiquidityWallet(address)._liquidity (MKFToken.sol#820) lacks a zero-check on :
- _liquidityReciever = _liquidity (MKFToken.sol#821)

variable: _liquidity
Source: MKFToken.sol
Lines: 820-820

node: _liquidityReciever = _liquidity
Source: MKFToken.sol
Lines: 821-821

Issue	missing-zero-check
Type	node
Impact	Low
Confidence	Medium
Name	pair = _address
Source	MKFToken.sol
Lines	843-843

MasterKeyToken.setLP(address)._address (MKFToken.sol#841) lacks a zero-check on :
- pair = _address (MKFToken.sol#843)

variable: _address
Source: MKFToken.sol
Lines: 841-841

node: pair = _address
Source: MKFToken.sol
Lines: 843-843

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

Name	_marketingWalletAddress = _marketing
Source	MKFToken.sol
Lines	813-813

MasterKeyToken.setMarketingWallet(address)._marketing (MKFToken.sol#812) lacks a zero-check on
:
- _marketingWalletAddress = _marketing (MKFToken.sol#813)

variable: _marketing
Source: MKFToken.sol
Lines: 812-812

node: _marketingWalletAddress = _marketing
Source: MKFToken.sol
Lines: 813-813

Issue	missing-zero-check
Type	node
Impact	Low
Confidence	Medium
Name	address(_receiver).transfer(balance)
Source	MKFToken.sol
Lines	801-801

MasterKeyToken.clearStuckBalance(address)._receiver (MKFToken.sol#799) lacks a zero-check on :
- address(_receiver).transfer(balance) (MKFToken.sol#801)

variable: _receiver
Source: MKFToken.sol
Lines: 799-799

node: address(_receiver).transfer(balance)
Source: MKFToken.sol
Lines: 801-801

Issue	missing-zero-check
Type	node
Impact	Low
Confidence	Medium
Name	BUSDDividendReceiver = _address
Source	MKFToken.sol
Lines	771-771

MasterKeyToken.setDistributor(address)._address (MKFToken.sol#769) lacks a zero-check on :
- BUSDDividendReceiver = _address (MKFToken.sol#771)

variable: _address
Source: MKFToken.sol
Lines: 769-769

node: BUSDDividendReceiver = _address
Source: MKFToken.sol
Lines: 771-771

Issue	missing-zero-check
Type	node
Impact	Low
Confidence	Medium
Name	deadWallet = _address
Source	MKFToken.sol
Lines	709-709

MasterKeyToken.setDeadWallet(address)._address (MKFToken.sol#708) lacks a zero-check on :
- deadWallet = _address (MKFToken.sol#709)

variable: _address
Source: MKFToken.sol
Lines: 708-708

node: deadWallet = _address
Source: MKFToken.sol
Lines: 709-709

Issue	calls-loop
Type	node
Impact	Low
Confidence	Medium
Name	BUSD.transfer(shareholder,amount)
Source	MKFToken.sol
Lines	285-285

DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290) has external calls inside a loop: BUSD.transfer(shareholder,amount) (MKFToken.sol#285)

function: distributeDividend
Source: MKFToken.sol

Lines: 279-290

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

Issue	reentrancy-benign
Type	node
Impact	Low
Confidence	Medium
Name	totalDividends = totalDividends.add(amount)
Source	MKFToken.sol
Lines	246-246

Reentrancy in DividendDistributor.deposit() (MKFToken.sol#235-248):
External calls:
- router.swapExactETHForTokensSupportingFeeOnTransferTokens{value:
msg.value}(0,path,address(this),block.timestamp) (MKFToken.sol#242)
State variables written after the call(s):
- dividendsPerShare =
dividendsPerShare.add(dividendsPerShareAccuracyFactor.mul(amount).div(totalShares))
(MKFToken.sol#247)
- totalDividends = totalDividends.add(amount) (MKFToken.sol#246)

function: deposit
Source: MKFToken.sol
Lines: 235-248

node: router.swapExactETHForTokensSupportingFeeOnTransferTokens{value:
msg.value}(0,path,address(this),block.timestamp)
Source: MKFToken.sol
Lines: 242-242

node: dividendsPerShare =
dividendsPerShare.add(dividendsPerShareAccuracyFactor.mul(amount).div(totalShares))
Source: MKFToken.sol
Lines: 247-247

node: totalDividends = totalDividends.add(amount)
Source: MKFToken.sol
Lines: 246-246

Issue	reentrancy-benign
Type	node

Impact	Low
Confidence	Medium
Name	_allowances[owner][spender] = amount
Source	MKFToken.sol
Lines	743-743

Reentrancy in MasterKeyToken.swapBack() (MKFToken.sol#648-686):
External calls:
- swapTokensForEth(tokensForSwap) (MKFToken.sol#664)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (MKFToken.sol#904-910)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
- addLiquidity(tokensForLP,amountETHLiquidity) (MKFToken.sol#684)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
External calls sending eth:
- address(_marketingWalletAddress).transfer(amountETHMarketing) (MKFToken.sol#675)
- address(_stakingWalletAddress).transfer(amountETHStaking) (MKFToken.sol#678)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
- addLiquidity(tokensForLP,amountETHLiquidity) (MKFToken.sol#684)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
State variables written after the call(s):
- addLiquidity(tokensForLP,amountETHLiquidity) (MKFToken.sol#684)
- _allowances[owner][spender] = amount (MKFToken.sol#743)

function: swapBack
Source: MKFToken.sol
Lines: 648-686

node: swapTokensForEth(tokensForSwap)
Source: MKFToken.sol
Lines: 664-664

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: MKFToken.sol
Lines: 904-910

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: MKFToken.sol
Lines: 684-684

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

Source: MKFToken.sol
Lines: 884-891

node: swapTokensForEth(tokensForSwap)
Source: MKFToken.sol
Lines: 664-664

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp)
Source: MKFToken.sol
Lines: 904-910

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: MKFToken.sol
Lines: 684-684

node: router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: MKFToken.sol
Lines: 684-684

node: _allowances[owner][spender] = amount
Source: MKFToken.sol
Lines: 743-743

Issue	reentrancy-benign
Type	node
Impact	Low
Confidence	Medium
Name	shareholderClaims[shareholder] = block.timestamp
Source	MKFToken.sol
Lines	286-286

Reentrancy in DividendDistributor.distributeDividend(address) (MKFToken.sol#279-290):
External calls:
- BUSD.transfer(shareholder,amount) (MKFToken.sol#285)
State variables written after the call(s):
- shareholderClaims[shareholder] = block.timestamp (MKFToken.sol#286)

function: distributeDividend
Source: MKFToken.sol
Lines: 279-290

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: shareholderClaims[shareholder] = block.timestamp
Source: MKFToken.sol
Lines: 286-286

Issue	reentrancy-benign
Type	node
Impact	Low
Confidence	Medium
Name	totalShares = totalShares.sub(shares[shareholder].amount).add(amount)
Source	MKFToken.sol
Lines	226-226

Reentrancy in DividendDistributor.setShare(address,uint256) (MKFToken.sol#215-229):
External calls:
- distributeDividend(shareholder) (MKFToken.sol#217)
- BUSD.transfer(shareholder,amount) (MKFToken.sol#285)
State variables written after the call(s):
- addShareholder(shareholder) (MKFToken.sol#221)
- shareholderIndexes[shareholder] = shareholders.length (MKFToken.sol#312)
- removeShareholder(shareholder) (MKFToken.sol#223)
- shareholderIndexes[shareholders[shareholders.length - 1]] = shareholderIndexes[shareholder] (MKFToken.sol#318)
- addShareholder(shareholder) (MKFToken.sol#221)
- shareholders.push(shareholder) (MKFToken.sol#313)
- removeShareholder(shareholder) (MKFToken.sol#223)
- shareholders[shareholderIndexes[shareholder]] = shareholders[shareholders.length - 1] (MKFToken.sol#317)
- shareholders.pop() (MKFToken.sol#319)
- totalShares = totalShares.sub(shares[shareholder].amount).add(amount) (MKFToken.sol#226)

function: setShare
Source: MKFToken.sol
Lines: 215-229

node: distributeDividend(shareholder)
Source: MKFToken.sol

Lines: 217-217

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: distributeDividend(shareholder)
Source: MKFToken.sol
Lines: 217-217

node: BUSD.transfer(shareholder,amount)
Source: MKFToken.sol
Lines: 285-285

node: addShareholder(shareholder)
Source: MKFToken.sol
Lines: 221-221

node: shareholderIndexes[shareholder] = shareholders.length
Source: MKFToken.sol
Lines: 312-312

node: removeShareholder(shareholder)
Source: MKFToken.sol
Lines: 223-223

node: shareholderIndexes[shareholders[shareholders.length - 1]] = shareholderIndexes[shareholder]
Source: MKFToken.sol
Lines: 318-318

node: addShareholder(shareholder)
Source: MKFToken.sol
Lines: 221-221

node: shareholders.push(shareholder)
Source: MKFToken.sol
Lines: 313-313

node: removeShareholder(shareholder)
Source: MKFToken.sol
Lines: 223-223

node: shareholders[shareholderIndexes[shareholder]] = shareholders[shareholders.length - 1]
Source: MKFToken.sol
Lines: 317-317

node: shareholders.pop()
Source: MKFToken.sol
Lines: 319-319

node: totalShares = totalShares.sub(shares[shareholder].amount).add(amount)
Source: MKFToken.sol

Lines: 226-226

Issue	reentrancy-events
Type	node
Impact	Low
Confidence	Medium
Name	Transfer(sender,recipient,AmountReceived)
Source	MKFToken.sol
Lines	613-613

Reentrancy in MasterKeyToken._transferFrom(address,address,uint256) (MKFToken.sol#580-615):
External calls:
- swapBack() (MKFToken.sol#593)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (MKFToken.sol#904-910)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
- distributor.setShare(sender,balanceOf(sender)) (MKFToken.sol#604)
- distributor.setShare(recipient,balanceOf(recipient)) (MKFToken.sol#605)
- marketDistributor.setShare(sender,balanceOf(sender)) (MKFToken.sol#608)
- marketDistributor.setShare(recipient,balanceOf(recipient)) (MKFToken.sol#609)
- distributor.process(distributorGas) (MKFToken.sol#611)
External calls sending eth:
- swapBack() (MKFToken.sol#593)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
- address(_marketingWalletAddress).transfer(amountETHMarketing) (MKFToken.sol#675)
- address(_stakingWalletAddress).transfer(amountETHStaking) (MKFToken.sol#678)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
Event emitted after the call(s):
- Transfer(sender,recipient,AmountReceived) (MKFToken.sol#613)

function: _transferFrom
Source: MKFToken.sol
Lines: 580-615

node: swapBack()
Source: MKFToken.sol
Lines: 593-593

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp)
Source: MKFToken.sol
Lines: 904-910

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: distributor.setShare(sender,balanceOf(sender))
Source: MKFToken.sol
Lines: 604-604

node: distributor.setShare(recipient,balanceOf(recipient))
Source: MKFToken.sol
Lines: 605-605

node: marketDistributor.setShare(sender,balanceOf(sender))
Source: MKFToken.sol
Lines: 608-608

node: marketDistributor.setShare(recipient,balanceOf(recipient))
Source: MKFToken.sol
Lines: 609-609

node: distributor.process(distributorGas)
Source: MKFToken.sol
Lines: 611-611

node: swapBack()
Source: MKFToken.sol
Lines: 593-593

node: router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: MKFToken.sol
Lines: 675-675

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: MKFToken.sol
Lines: 678-678

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: Transfer(sender,recipient,AmountReceived)
Source: MKFToken.sol

Lines: 613-613

Issue	reentrancy-events
Type	node
Impact	Low
Confidence	Medium
Name	addLiquidity(tokensForLP,amountETHLiquidity)
Source	MKFToken.sol
Lines	684-684

Reentrancy in MasterKeyToken.swapBack() (MKFToken.sol#648-686):
External calls:
- swapTokensForEth(tokensForSwap) (MKFToken.sol#664)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (MKFToken.sol#904-910)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
- addLiquidity(tokensForLP,amountETHLiquidity) (MKFToken.sol#684)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
External calls sending eth:
- address(_marketingWalletAddress).transfer(amountETHMarketing) (MKFToken.sol#675)
- address(_stakingWalletAddress).transfer(amountETHStaking) (MKFToken.sol#678)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
- addLiquidity(tokensForLP,amountETHLiquidity) (MKFToken.sol#684)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
Event emitted after the call(s):
- Approval(owner,spender,amount) (MKFToken.sol#744)
- addLiquidity(tokensForLP,amountETHLiquidity) (MKFToken.sol#684)

function: swapBack
Source: MKFToken.sol
Lines: 648-686

node: swapTokensForEth(tokensForSwap)
Source: MKFToken.sol
Lines: 664-664

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: MKFToken.sol
Lines: 904-910

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol

Lines: 681-681

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: MKFToken.sol
Lines: 684-684

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: MKFToken.sol
Lines: 675-675

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: MKFToken.sol
Lines: 678-678

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: MKFToken.sol
Lines: 684-684

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: Approval(owner,spender,amount)
Source: MKFToken.sol
Lines: 744-744

node: addLiquidity(tokensForLP,amountETHLiquidity)
Source: MKFToken.sol
Lines: 684-684

Issue	reentrancy-events
Type	node
Impact	Low
Confidence	Medium
Name	AmountReceived = takeFee(sender,recipient,amount)
Source	MKFToken.sol

Lines	598-600

Reentrancy in MasterKeyToken._transferFrom(address,address,uint256) (MKFToken.sol#580-615):
External calls:
- swapBack() (MKFToken.sol#593)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
- router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this)
,block.timestamp) (MKFToken.sol#904-910)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
External calls sending eth:
- swapBack() (MKFToken.sol#593)
- router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
(MKFToken.sol#884-891)
- address(_marketingWalletAddress).transfer(amountETHMarketing) (MKFToken.sol#675)
- address(_stakingWalletAddress).transfer(amountETHStaking) (MKFToken.sol#678)
- distributor.deposit{value: amountETHReward}() (MKFToken.sol#681)
Event emitted after the call(s):
- Transfer(sender,address(this),feeAmount) (MKFToken.sol#635)
- AmountReceived = takeFee(sender,recipient,amount) (MKFToken.sol#598-600)
- Transfer(sender,address(deadWallet),BFEE) (MKFToken.sol#640)
- AmountReceived = takeFee(sender,recipient,amount) (MKFToken.sol#598-600)

function: _transferFrom
Source: MKFToken.sol
Lines: 580-615

node: swapBack()
Source: MKFToken.sol
Lines: 593-593

node: router.addLiquidityETH{value:
ethAmount}(address(this),tokenAmount,0,0,_liquidityReciever,block.timestamp)
Source: MKFToken.sol
Lines: 884-891

node: router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address
(this),block.timestamp)
Source: MKFToken.sol
Lines: 904-910

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: swapBack()
Source: MKFToken.sol
Lines: 593-593

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

Source: MKFToken.sol
Lines: 884-891

node: address(_marketingWalletAddress).transfer(amountETHMarketing)
Source: MKFToken.sol
Lines: 675-675

node: address(_stakingWalletAddress).transfer(amountETHStaking)
Source: MKFToken.sol
Lines: 678-678

node: distributor.deposit{value: amountETHReward}()
Source: MKFToken.sol
Lines: 681-681

node: Transfer(sender,address(this),feeAmount)
Source: MKFToken.sol
Lines: 635-635

node: AmountReceived = takeFee(sender,recipient,amount)
Source: MKFToken.sol
Lines: 598-600

node: Transfer(sender,address(deadWallet),BFEE)
Source: MKFToken.sol
Lines: 640-640

node: AmountReceived = takeFee(sender,recipient,amount)
Source: MKFToken.sol
Lines: 598-600

Issue	timestamp
Type	node
Impact	Low
Confidence	Medium
Name	shareholderClaims[shareholder] + minPeriod < block.timestamp && getUnpaidEarnings(shareholder)
Source	MKFToken.sol
Lines	276-276

DividendDistributor.shouldDistribute(address) (MKFToken.sol#275-277) uses timestamp for comparisons
Dangerous comparisons:
- shareholderClaims[shareholder] + minPeriod < block.timestamp && getUnpaidEarnings(shareholder) > minDistribution (MKFToken.sol#276)

function: shouldDistribute
Source: MKFToken.sol

Lines: 275-277

node: shareholderClaims[shareholder] + minPeriod < block.timestamp &&
getUnpaidEarnings(shareholder) > minDistribution
Source: MKFToken.sol
Lines: 276-276