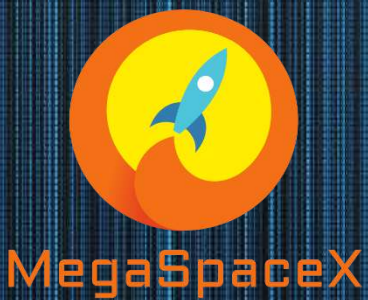
RugFreeCoins Audit



MegaSpaceX Token

Smart Contract Security Audit

July 23, 2021

Contents

Audit details	1
Disclaimer	2
Background	3
About the project	4
Target market and the concept	5
Potential to grow with score points	7
Total Points	7
Contract details	8
Token distribution	9
Contract code function details	10
Contract description table	11
Security issue checking status	21
Owner privileges	22
Audit conclusion	25

Audit details

Audited project

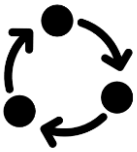
MegaspaceX Token

Contract Address

0xa8501b9e22d9177d498da4852c9c0acff11eb791

Client contact

MegaspaceX Team



Blockchain

Binance smart chain

Project website

https://www.megaspacex.eu/

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Rugfreecoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Rugfreecoins) owe no duty of care towards you or any other person, nor does Rugfreecoins make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Rugfreecoins hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Rugfreecoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Rugfreecoins, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

Rugfreecoins was commissioned by MegaspaceX to perform an audit of the smart contract.

https://bscscan.com/address/0xa8501b9e22d9177d498da4852c9c0acff11eb791

The focus of this audit is to verify that the smart contract is secure, resilient and working according to the specifications.

The information in this report should be used to understand the risk exposure of the smart contract, project feasibility, long term sustainability and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

About the project

MegaSpaceX ($MSPX) is a completely innovative Coin based on Binance Smart Chain that integrates,

- One-way tax
- Auto-claim BNB rewards
- Auto-buyback Boost
- Anti-dump and Auto-LP
- Anti-bot Exploit

MegaSpaceX is designed to bring a constant and stable price-increasing mechanism for the tokens, and with its unique and powerful features, MegasSpaceX will guarantee to bring users to mars!

MegaSpaceX project aims to contribute to the research of interstellar travel and mars immigration and eventually make humanity a multi-planet species. This coincides with Elon Musk's ideas.

The community of MSPX will benefit from the value increasing of the token, meanwhile, contributing to the future of mankind.

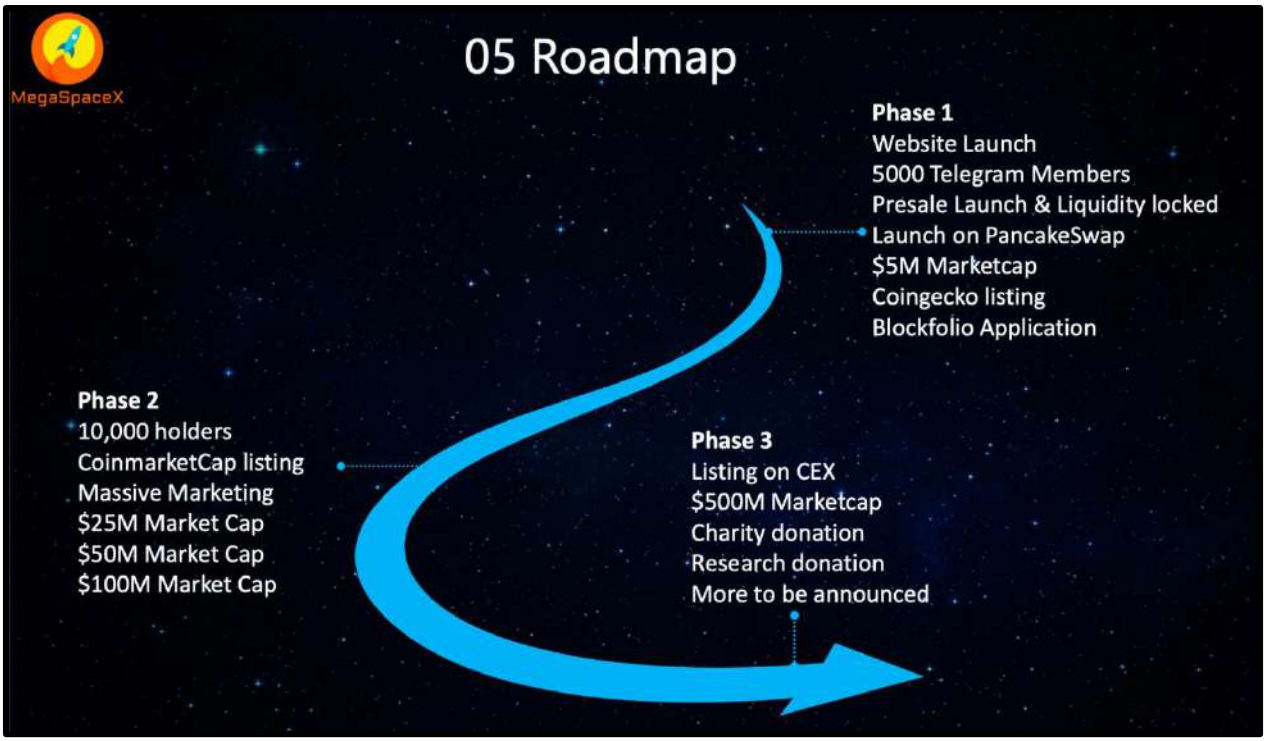
Tokenomics

0% tax when buying

20% tax when selling

- 10% of every trade goes to holders pockets in BNB.
- 5% of every trade goes to the liquidity pool.
- 2.5% of every trade goes to the marketing wallet.
- 2.5% of every trade goes to a charity wallet.

Roadmap



Target market and the concept

Target market

- Anyone who's interested in Crypto space with long term investment plans.
- Anyone who's ready to earn a passive income by holding tokens.
- Anyone who's interested in trading tokens.
- Anyone who's interested in making financial transactions with any other party using megaspaceX as the currency.
- Anyone who's interested in contributing to the research of interstellar travel and mars immigration and eventually makes humanity a multi-planet species by taking part with the project.

The BNB reward system

10% of each transaction gets converted to BNBs and is split amongst all holders. The rewards are sent to holders that have at least 10,000 **MSPX** tokens, holders will be eligible to receive tokens everyone hour and rewards are proportional to how many tokens each individual hold.

Sustainable mechanism

The liquidity fee of 5%, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

The **fee of 2.5% marketing** is what allows megaspaceX to promote the token and use funds to further development of the platform. Tokens will be swapped into BNBs and will be sent to a marketing wallet per transaction. This way, megaspaceX will have access to the funds without selling tokens as the traditional way, which will enable them to consume funds without hurting the project.

The buyback and burn mechanism collect 2.5% tax on each transaction, which is stored inside the contract. Whenever a sell occurs, a fraction of the buyback amount is used to automatically purchase tokens from the liquidity pool. Those tokens are immediately burned after purchase, which keeps the token price stable.

Anti-dumping and anti-whale strategy

The megaspaceX contract includes a function that stops all sales above 0.1% of the total supply. This will discourage (mini)-whales from dumping all their bags at once.

Anti pump and dump: Certain groups of individuals practice pump and dump schemes, in order to lure in outside investors by the looks of a bullish chart, and sell at a high point.megaspaceX charges 20% one way fee when selling.

Anti bot exploit system

The Buyback Boost will only cover the same sell value under a certain threshold. So massive bot-triggered micro-sells will not dry out the buyback fund.

The future plan

The MegaSpaceX project aims to contribute to the research of interstellar travel and mars immigration and eventually makes humanity a multi-planet species. This coincides with Elon Musk's ideas.

The project’s community firmly believes that humanity will not and should not stay in the cradle forever. More and more people will realize the importance of humanity being able to inhabit multiple planets and make interstellar travel.

With the strong support from the community and by cooperating with NASA, SpaceX, and other research institutions all over the world, MegaSpaceX will work as a boost for space exploration. MSPX holders and traders are therefore contributing to the future of whole humanity.

Potential to grow with score points

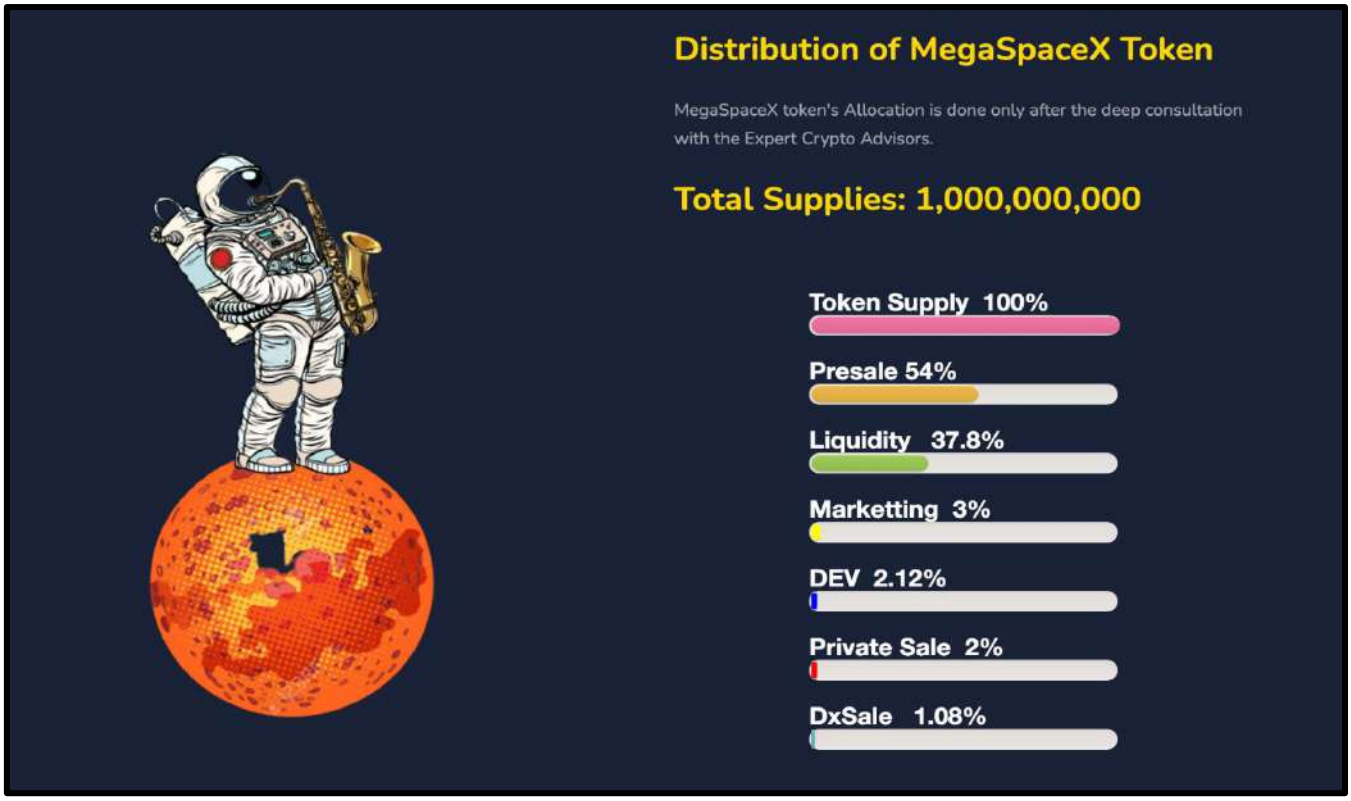
1.	Project efficiency	7/10
2.	Project uniqueness	7/10
3	Information quality	8/10
4	Service quality	8/10
5	System quality	7/10
6	Impact on the community	8/10
7	Impact on the business	8/10
8	Preparing for the future	6/10
Total Points		**7.38/10**

Contract details

Token contract details for 23rd July 2021

Contract name	MegaspaceX Token
Contract address	0xa8501b9e22d9177d498da4852c9c0acff11eb791
Token supply	1,000,000,000
Token ticker	MSPX
Decimals	18
Token holders	1(Prior to the launch)
Transaction count	1 (Prior to the launch)
Top 100% holders dominance	
Contract deployer address	0x9276107F2d98B672D64a32909Ab67C33168B4485
Contract's current owner address	0x9276107F2d98B672D64a32909Ab67C33168B4485

Token distribution



Contract code function details

No	Category	Item	Result
1	Coding conventions	BRC20 Token standards	pass
		compile errors	pass
		Compiler version security	pass
		visibility specifiers	pass
		Gas consumption	low issue
		SafeMath features	pass
		Fallback usage	pass
		tx.origin usage	pass
		deprecated items	pass
		Redundant code	pass
		Overriding variables	pass
2	Function call audit	Authorization of function call	pass
		Low level function (call/delegate call) security	pass
		Returned value security	pass
		Selfdestruct function security	pass
3	Business security	Access control of owners	pass
		Business logics	pass
		Business implementations	pass
4	Integer overflow/underflow		pass
5	Reentrancy		pass
6	Exceptional reachable state		pass
7	Transaction ordering dependence		pass
8	Block properties dependence		pass
9	Pseudo random number generator (PRNG)		pass
10	DoS (Denial of Service)		pass
11	Token vesting implementation		pass
12	Fake deposit		pass
13	Event security		pass

Contract description table

Below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions and implementations with its visibility and mutability.

Contract	**Type**	**Bases**		
L	Function Name	Visibility	Mutability	Modifiers
MegaSpaceX	**Implementation**	**ERC20, Ownable**		
L	buyBackUpperLimitAmount	Public ❗		NO❗
L	setBuybackUpperLimit	External ❗	🛑	onlyOwner
L	setBuyBackEnabled	Public ❗	🛑	onlyOwner
L		Public ❗	🛑	ERC20
L		External ❗	💵	NO❗
L	updateDividendTracker	Public ❗	🛑	onlyOwner
L	updateUniswapV2Router	Public ❗	🛑	onlyOwner
L	excludeFromFees	Public ❗	🛑	onlyOwner
L	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
L	_setAutomatedMarketMakerPair	Private 🔐	🛑	
L	setfees	External ❗	🛑	onlyOwner
L	setBuybackindex	External ❗	🛑	onlyOwner
L	setMarketingAddress	External ❗	🛑	onlyOwner
L	updateLiquidityWallet	Public ❗	🛑	onlyOwner

L	updateGasForProcessing	Public ❗	🛑	onlyOwner
L	updateClaimWait	External ❗	🛑	onlyOwner
L	getClaimWait	External ❗		NO❗
L	getTotalDividendsDistributed	External ❗		NO❗
L	isExcludedFromFees	Public ❗		NO❗
L	withdrawableDividendOf	Public ❗		NO❗
L	dividendTokenBalanceOf	Public ❗		NO❗
L	getAccountDividendsInfo	External ❗		NO❗
L	getAccountDividendsInfoAtIndex	External ❗		NO❗
L	processDividendTracker	External ❗	🛑	NO❗
L	claim	External ❗	🛑	NO❗
L	getLastProcessedIndex	External ❗		NO❗
L	_transfer	Internal 🔒	🛑	
L	transfertomarketwallet	Private 🔐	🛑	
L	transferToAddressETH	Private 🔐	🛑	
L	swapAndLiquify	Private 🔐	🛑	
L	swapTokensForEth	Private 🔐	🛑	
L	addLiquidity	Private 🔐	🛑	
L	swapAndSendDividends	Private 🔐	🛑	
L	_getTokenSwapRate	Private 🔐		

L	buyBackTokens	Private 🔐	🛑	
L	swapETHForTokens	Private 🔐	🛑	
L	afterPreSale	External ❗	🛑	onlyOwner
MSPXDividendTracker	**Implementation**	**DividendPayingToken, Ownable**		
L		Public ❗	🛑	DividendPayingToken
L	_transfer	Internal 🔒	🛑	
L	withdrawDividend	Public ❗	🛑	NO❗
L	excludeFromDividends	External ❗	🛑	onlyOwner
L	updateClaimWait	External ❗	🛑	onlyOwner
L	getLastProcessedIndex	External ❗		NO❗
L	getNumberOfTokenHolders	External ❗		NO❗
L	getAccount	Public ❗		NO❗
L	getAccountAtIndex	Public ❗		NO❗
L	canAutoClaim	Private 🔐		
L	setBalance	External ❗	🛑	onlyOwner
L	process	Public ❗	🛑	NO❗
L	processAccount	Public ❗	🛑	onlyOwner
Context	**Implementation**			
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		

DividendPayingToken	**Implementation**	**ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface**		
L		Public ❗	🛑	ERC20
L		External ❗	💵	NO❗
L	distributeDividends	Public ❗	💵	NO❗
L	withdrawDividend	Public ❗	🛑	NO❗
L	_withdrawDividendOfUser	Internal 🔒	🛑	
L	dividendOf	Public ❗		NO❗
L	withdrawableDividendOf	Public ❗		NO❗
L	withdrawnDividendOf	Public ❗		NO❗
L	accumulativeDividendOf	Public ❗		NO❗
L	_transfer	Internal 🔒	🛑	
L	_mint	Internal 🔒	🛑	
L	_burn	Internal 🔒	🛑	
L	_setBalance	Internal 🔒	🛑	
DividendPayingTokenInterface	**Interface**			
L	dividendOf	External ❗		NO❗
L	distributeDividends	External ❗	💵	NO❗
L	withdrawDividend	External ❗	🛑	NO❗

DividendPayingTokenOptionalInterface	Interface			
L	withdrawableDividendOf	External ❗		NO❗
L	withdrawnDividendOf	External ❗		NO❗
L	accumulativeDividendOf	External ❗		NO❗
ERC20	**Implementation**	**Context, IERC20, IERC20Metadata**		
L		Public ❗	🛑	NO❗
L	name	Public ❗		NO❗
L	symbol	Public ❗		NO❗
L	decimals	Public ❗		NO❗
L	totalSupply	Public ❗		NO❗
L	balanceOf	Public ❗		NO❗
L	transfer	Public ❗	🛑	NO❗
L	allowance	Public ❗		NO❗
L	approve	Public ❗	🛑	NO❗
L	transferFrom	Public ❗	🛑	NO❗
L	increaseAllowance	Public ❗	🛑	NO❗
L	decreaseAllowance	Public ❗	🛑	NO❗
L	_transfer	Internal 🔒	🛑	
L	_mint	Internal 🔒	🛑	
L	_burn	Internal 🔒	🛑	
L	_approve	Internal 🔒	🛑	
L	_beforeTokenTransfer	Internal 🔒	🛑	

IERC20	**Interface**			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC20Metadata	**Interface**	**IERC20**		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
IterableMapping	**Library**			
└	get	Public ❗		NO❗
└	getIndexOfKey	Public ❗		NO❗
└	getKeyAtIndex	Public ❗		NO❗
└	size	Public ❗		NO❗
└	set	Public ❗	🛑	NO❗
└	remove	Public ❗	🛑	NO❗
IUniswapV2Factory	**Interface**			
└	feeTo	External ❗		NO❗
└	feeToSetter	External ❗		NO❗
└	getPair	External ❗		NO❗
└	allPairs	External ❗		NO❗

└	allPairsLength	External ❗		NO❗
└	createPair	External ❗	🛑	NO❗
└	setFeeTo	External ❗	🛑	NO❗
└	setFeeToSetter	External ❗	🛑	NO❗
IUniswapV2Pair	**Interface**			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🛑	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗

L	price1CumulativeLast	External ❗		NO❗
L	kLast	External ❗		NO❗
L	mint	External ❗	🛑	NO❗
L	burn	External ❗	🛑	NO❗
L	swap	External ❗	🛑	NO❗
L	skim	External ❗	🛑	NO❗
L	sync	External ❗	🛑	NO❗
L	initialize	External ❗	🛑	NO❗
IUniswapV2Router01	**Interface**			
L	factory	External ❗		NO❗
L	WETH	External ❗		NO❗
L	addLiquidity	External ❗	🛑	NO❗
L	addLiquidityETH	External ❗	💵	NO❗
L	removeLiquidity	External ❗	🛑	NO❗
L	removeLiquidityETH	External ❗	🛑	NO❗
L	removeLiquidityWithPermit	External ❗	🛑	NO❗
L	removeLiquidityETHWithPermit	External ❗	🛑	NO❗
L	swapExactTokensForTokens	External ❗	🛑	NO❗
L	swapTokensForExactTokens	External ❗	🛑	NO❗
L	swapExactETHForTokens	External ❗	💵	NO❗
L	swapTokensForExactETH	External ❗	🛑	NO❗

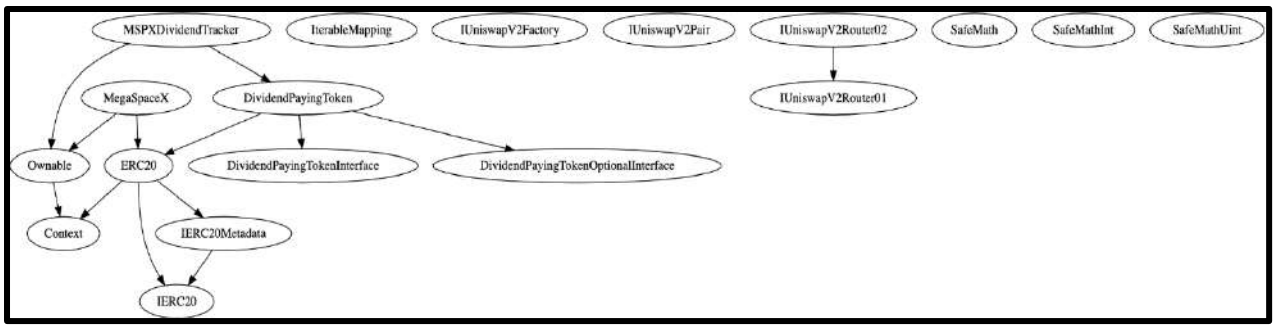
L	swapExactTokens ForETH	External ❗	🛑	NO❗
L	swapETHForExact Tokens	External ❗	💵	NO❗
L	quote	External ❗		NO❗
L	getAmountOut	External ❗		NO❗
L	getAmountIn	External ❗		NO❗
L	getAmountsOut	External ❗		NO❗
L	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	**Interface**	**IUniswapV2Route r01**		
L	removeLiquidityET HSupportingFeeO nTransferTokens	External ❗	🛑	NO❗
L	removeLiquidityET HWithPermitSupp ortingFeeOnTransf erTokens	External ❗	🛑	NO❗
L	swapExactTokens ForTokensSupport ingFeeOnTransfer Tokens	External ❗	🛑	NO❗
L	swapExactETHFor TokensSupporting FeeOnTransferTo kens	External ❗	💵	NO❗
L	swapExactTokens ForETHSupporting FeeOnTransferTo kens	External ❗	🛑	NO❗
Ownable	**Implementation**	**Context**		
L		Public ❗	🛑	NO❗
L	owner	Public ❗		NO❗

└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
SafeMath	**Library**			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
SafeMathInt	**Library**			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	**Library**			
└	toInt256Safe	Internal 🔒		

Legend

Symbol	Meaning
	Function can modify state
	**Function is payable**

Inheritance Hierarchy



Security issue checking status

- **High severity issues**
  No high severity issues found.

  .

- **Medium severity issues**
  No medium severity issues found.

❖ **Low severity issues**

➢ **BNB transfers with a low gas limit.**

_withdrawDividendOfUser() function of DividendPayingToken contract transfers ETH/BNB via .call{value, gas} method with 3000 of gas limit. This relatively small constant may cause problems with future ETH/BSC updates as operation gas costs may change with forks.

```
ftrace | funcSig
function _withdrawDividendOfUser(address payable user⬆) internal returns (uint256) {
  uint256 _withdrawableDividend = withdrawableDividendOf(user⬆);
  if (_withdrawableDividend > 0) {
    withdrawnDividends[user⬆] = withdrawnDividends[user⬆].add(_withdrawableDividend);
    emit DividendWithdrawn(user⬆, _withdrawableDividend);
    (bool success,) = user⬆.call{value: _withdrawableDividend, gas: 3000}("");

    if(!success) {
      withdrawnDividends[user⬆] = withdrawnDividends[user⬆].sub(_withdrawableDividend);
      return 0;
    }

    return _withdrawableDividend;
  }

  return 0;
}
```

Owner privileges

❖ The owner can enable/disable buyback function.

```
ftrace | funcSig
function setBuyBackEnabled(bool _enabled⬆) public onlyOwner {
    buyBackEnabled = _enabled⬆;
    emit BuyBackEnabledUpdated(_enabled⬆);
}
```

❖ The owner can change buyback upper limit.

```
ftrace | funcSig
function setBuybackUpperLimit(uint256 buyBackLimit⬆) external onlyOwner() {
    buyBackUpperLimit = buyBackLimit⬆ * 10**18;
}

ftrace | funcSig
```

❖ The owner can update pancake swap router.

```
ftrace | funcSig
function updateUniswapV2Router(address newAddress⬆) public onlyOwner {
    require(
        newAddress⬆ != address(uniswapV2Router),
        "MSPX: The router already has that address"
    );
    emit UpdateUniswapV2Router(newAddress⬆, address(uniswapV2Router));
    uniswapV2Router = IUniswapV2Router02(newAddress⬆);
}
```

❖ The owner can exclude wallets from fees.

```
ftrace | funcSig
function excludeFromFees(address account⬆, bool excluded⬆) public onlyOwner {
    require(
        _isExcludedFromFees[account⬆] != excluded⬆,
        "MSPX: Account is already the value of 'excluded'"
    );
    _isExcludedFromFees[account⬆] = excluded⬆;

    emit ExcludeFromFees(account⬆, excluded⬆);
}
```

❖ The owner can change marketing address.

```
ftrace | funcSig
function setMarketingAddress(address _marketingAddress⬆)
    external
    onlyOwner()
{
    marketingAddress = payable(_marketingAddress⬆);
}
```

❖ The owner can update the liquidity wallet.

```
ftrace | funcSig
function updateLiquidityWallet(address newLiquidityWallet⬆)
    public
    onlyOwner
{
    require(
        newLiquidityWallet⬆ != liquidityWallet,
        "Token: liquidity wallet is already this address"
    );
    excludeFromFees(newLiquidityWallet⬆, true);
    emit LiquidityWalletUpdated(newLiquidityWallet⬆, liquidityWallet);
    liquidityWallet = newLiquidityWallet⬆;
}
```

❖ The owner can update the gas fee.

```
ftrace | funcSig
function updateGasForProcessing(uint256 newValue⬆) public onlyOwner {
    require(
        newValue⬆ >= 200000 && newValue⬆ <= 500000,
        "MSPX: gasForProcessing must be between 200,000 and 500,000"
    );
    require(
        newValue⬆ != gasForProcessing,
        "MSPX: Cannot update gasForProcessing to same value"
    );
    emit GasForProcessingUpdated(newValue⬆, gasForProcessing);
    gasForProcessing = newValue⬆;
}
```

❖ The owner can change the BNB claim wait time.

```
ftrace | funcSig
function updateClaimWait(uint256 claimWait⬆) external onlyOwner {
    dividendTracker.updateClaimWait(claimWait⬆);
}
```

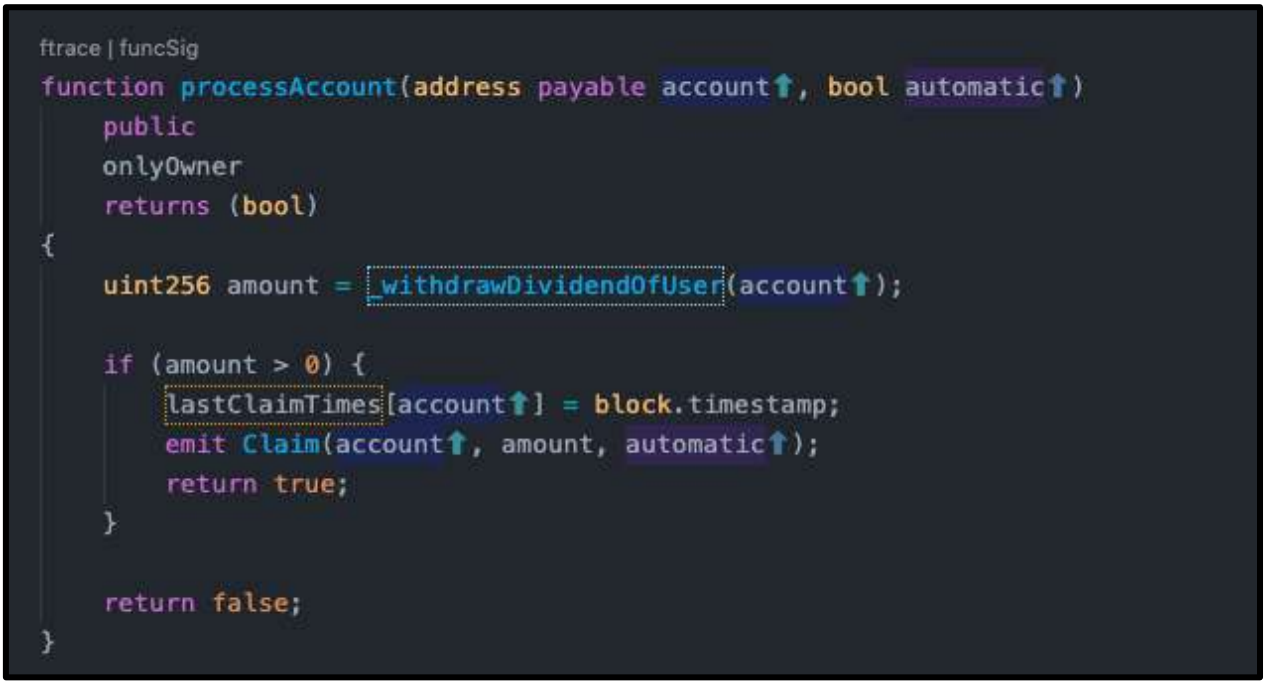
❖ The owner can exclude accounts from dividends.

```
ftrace | funcSig
function excludeFromDividends(address account⬆) external onlyOwner {
    require(!excludedFromDividends[account⬆]);
    excludedFromDividends[account⬆] = true;

    _setBalance(account⬆, 0);
    tokenHoldersMap.remove(account⬆);

    emit ExcludeFromDividends(account⬆);
}
```

❖ The owner can manually process dividends to wallets.



```
ftrace | funcSig
function processAccount(address payable account⬆, bool automatic⬆)
    public
    onlyOwner
    returns (bool)
{
    uint256 amount = _withdrawDividendOfUser(account⬆);

    if (amount > 0) {
        lastClaimTimes[account⬆] = block.timestamp;
        emit Claim(account⬆, amount, automatic⬆);
        return true;
    }

    return false;
}
```

Audit conclusion

While conducting the audit of the megaspaceX token smart contract, it was observed that there is nothing alamaring with the code and the contract contains only a low severity issue.