



SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



Meta Space Swap
\$Meta Space



08/04/2022



TABLE OF CONTENTS

- 1 DISCLAIMER
- 2 INTRODUCTION
- 3-4 AUDIT OVERVIEW
- 5-6 OWNER PRIVILEGES
- 7 CONCLUSION AND ANALYSIS
- 8 TOKEN DETAILS
- 9 META SPACE SWAP TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS
- 10 TECHNICAL DISCLAIMER



DISCLAIMER

The information provided on this analysis document is only for general information and should not be used as a reason to invest.

FreshCoins Team will take no payment for manipulating the results of this audit.

The score and the result will stay on this project page information on our website <https://freshcoins.io>

FreshCoins Team does not guarantees that a project will not sell off team supply, or any other scam strategy (RUG or Honeypot etc)



INTRODUCTION

FreshCoins (Consultant) was contracted by
Meta Space Swap (Customer) to conduct a Smart Contract Code Review
and Security Analysis.

0x6527E43a5f9e3d7bdBe0e717F2AaF1206E545307

Network: Binance Smart Chain (BSC)

This report presents the findings of the security assessment of
Customer's smart contract and its code review conducted on 08/04/2022



AUDIT OVERVIEW



Security Score



Static Scan
Automatic scanning for common vulnerabilities



ERC Scan
Automatic checks for ERC's conformance

0 High

0 Medium

1 Low

0 Optimizations

0 Informational



No.	Issue description	Checking Status
1	Compiler Errors / Warnings	Passed
2	Reentrancy and Cross-function	Passed
3	Front running	Passed
4	Timestamp dependence	Passed
5	Integer Overflow and Underflow	Passed
6	Reverted DoS	Passed
7	DoS with block gas limit	Low
8	Methods execution permissions	Passed
9	Exchange rate impact	Passed
10	Malicious Event	Passed
11	Scoping and Declarations	Passed
12	Uninitialized storage pointers	Passed
13	Design Logic	Passed
14	Safe Zeppelin module	Passed

OWNER PRIVILEGES

Contract owner can't mint tokens after initial contract deploy

Contract owner can exclude/include wallet from tax

```
function excludeFromFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = true;
}

function includeInFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = false;
}
```

Contract owner can burn tokens from a specific wallet

```
function _burn(address account, uint256 amount) internal virtual {
    require(account != address(0), "ERC20: burn from the zero address");

    _beforeTokenTransfer(account, address(0), amount);

    uint256 accountBalance = _balances[account];
    require(accountBalance >= amount, "ERC20: burn amount exceeds balance");
    unchecked {
        _balances[account] = accountBalance - amount;
    }
    _totalSupply -= amount;

    emit Transfer(account, address(0), amount);
    _afterTokenTransfer(account, address(0), amount);
}

function burnFrom(address account, uint256 amount) public virtual {
    uint256 currentAllowance = allowance(account, _msgSender());
    require(currentAllowance >= amount, "ERC20: burn amount exceeds allowance");
    unchecked {
        _approve(account, _msgSender(), currentAllowance - amount);
    }
    _burn(account, amount);
}
```

Contract owner can change max tx amount

```
function SetMaxTxAmount(uint256 newMaxAmount) external onlyOwner {
    _maxTxAmount = newMaxAmount;
}
```

Contract owner can change fees up to 100%

```
function SetBuyValues(uint lp,uint market, uint burn) external onlyOwner {
    LPBuyFees = lp;
    marketBuyFees = market;
    burnBuyFees = burn;
}

function SetSellValues(uint lp,uint market, uint burn) external onlyOwner {
    LPSellFees = lp;
    marketSellFees = market;
    burnSellFees = burn;
}
```

Contract owner can change `addressForMarketing` address

Current value:

`addressForMarketing` : [0xa5b52c1befc099fbe522e10efd2e072590708cf1](#)

```
function SetAddressForMarketing(address newAddress) external onlyOwner {
    addressForMarketing = newAddress;
}
```

Contract owner can transfer tokens to another smart contract

```
function SetMintContract(address contrat_addr) external onlyOwner {
    super._transfer(mintContract,contrat_addr,balanceOf(address(this)));
    mintContract = contrat_addr;
}
```

Contract owner can change swap settings

```
function SetSwapAndLiquifyEnabled(bool isLpSwap) external onlyOwner {
    swapAndLiquifyEnabled = isLpSwap;
}

function SetSwapSellLiquifyEnabled(bool isLpSwap) external onlyOwner {
    swapSellLiquifyEnabled = isLpSwap;
}

function SetSwapBuyLiquifyEnabled(bool isLpSwap) external onlyOwner {
    swapBuyLiquifyEnabled = isLpSwap;
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
    _setOwner(address(0));
}
```

Contract owner can transfer ownership

```
function transferOwnership(address newOwner) public virtual onlyOwner {
    require(newOwner != address(0), "Ownable: new owner is the zero address");
    _setOwner(newOwner);
}
```

CONCLUSION AND ANALYSIS



Smart Contracts within the scope were manually reviewed and analyzed with static tools.



Audit report overview contains all found security vulnerabilities and other issues in the reviewed code.



Found 1 LOW issue during the first review.

TOKEN DETAILS

Details

Buy fees:	10%
Sell fees:	12%
Max TX:	50,000,000
Max Sell:	N/A

Honeypot Risk

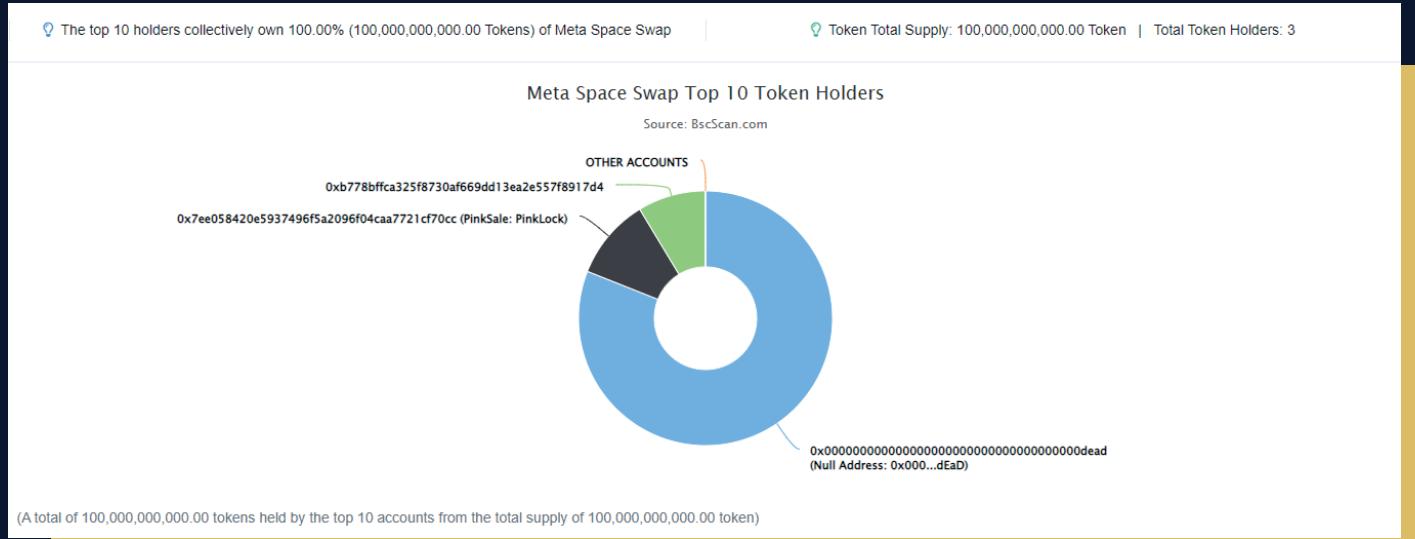
Ownership:	Owned
Blacklist:	Not detected
Modify Max TX:	Detected
Modify Max Sell:	Not detected
Disable Trading:	Not detected

Rug Pull Risk

Liquidity:	N/A
Holders:	Clean



META SPACE SWAP TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS



Rank	Address	Quantity (Token)	Percentage
1	Null Address: 0x000..dEaD	81,000,000,000	81.0000%
2	PinkSale: PinkLock	10,355,500,000	10.3555%
3	0xb778bffca325f8730af669dd13ea2e557f8917d4	8,644,500,000	8.6445%

TECHNICAL DISCLAIMER

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. The audit can't guarantee the explicit security of the audited project / smart contract.

