



SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



SuperPancake
\$SUPERCake

28/06/2022

TABLE OF CONTENTS

- 1 **DISCLAIMER**
- 2 **INTRODUCTION**
- 3-4 **AUDIT OVERVIEW**
- 5-7 **OWNER PRIVILEGES**
- 8 **CONCLUSION AND ANALYSIS**
- 9 **TOKEN DETAILS**
- 10 **SUPERPANCAKE TOKEN ANALYTICS &
TOP 10 TOKEN HOLDERS**
- 11 **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 Honeygot etc)



INTRODUCTION

FreshCoins (Consultant) was contracted by **SuperPancake** (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0x2f3b3400427C359F9E4559C4F41C6e6e2D254ACa

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 **28/06/2022**



AUDIT OVERVIEW



Security Score



Static Scan

Automatic scanning for common vulnerabilities



ERC Scan

Automatic checks for ERC's conformance



High



Medium



Low



Optimizations



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	Passed
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't exclude an address from transactions

Contract owner can burn tokens

```
function burn(uint256 amount) external {
    _burn(msg.sender, amount);
}

function _burn(address account, uint256 amount) internal {
    require(amount != 0);
    require(amount <= _balances[account]);
    _balances[account] = _balances[account].sub(amount);
    _totalSupply = _totalSupply.sub(amount);
    emit Transfer(account, address(0), amount);
}
```

Contract owner can exclude/include wallet from tax

```
function setIsFeeExempt(address holder, bool exempt) external authorized {
    isFeeExempt[holder] = exempt;
}
```

Contract owner can exclude/include wallet from dividends

```
function setIsDividendExempt(address holder, bool exempt) external authorized {
    require(holder != address(this) && holder != pair);
    isDividendExempt[holder] = exempt;
    if(exempt){
        distributor.setShare(holder, 0);
    }else{
        distributor.setShare(holder, _balances[holder]);
    }
}
```

Contract owner can exclude/include wallet from tx limitations

```
function setIsTxLimitExempt(address holder, bool exempt) external authorized {
    isTxLimitExempt[holder] = exempt;
}
```

Contract owner can exclude/include wallet from tx cooldown

```
function setIsTimelockExempt(address holder, bool exempt) external authorized {
    isTimelockExempt[holder] = exempt;
}
```

Contract owner can change max wallet amount (without threshold)

```
function setMaxWallet(uint256 maxWalletPercent) public onlyOwner {
    _maxWallet = maxWalletPercent;
}
```

Contract owner can change max tx amount (with threshold)

```
function setTxLimit(uint256 amount) external authorized {
    require(amount >= _totalSupply / 1000);
    _maxTxAmount = amount;
}
```

Contract owner can change swap settings

```
function setSwapBackSettings(bool _enabled, uint256 _amount) external authorized {
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

Contract owner can change fees up to 100%

```
function setFees(uint256 _liquidityFee, uint256 _buybackFee, uint256 _reflectionFee, uint256 _marketingdevelopmentFee, uint256 _burnFee, uint256 _extraFeeOnSell, uint256 _feeDenominator) external authorized {
    liquidityFee = _liquidityFee;
    buybackFee = _buybackFee;
    reflectionFee = _reflectionFee;
    marketingdevelopmentFee = _marketingdevelopmentFee;
    burnFee = _burnFee;
    totalFee = _liquidityFee.add(_buybackFee).add(_reflectionFee).add(_marketingdevelopmentFee);
    extraFeeOnSell = _extraFeeOnSell;
    feeDenominator = _feeDenominator;
}
```

Contract owner can change developmentReceiver and marketingFeeReceiver addresses

```
function setFeeReceivers(address _developmentReceiver, address _marketingFeeReceiver) external authorized {
    developmentReceiver = _developmentReceiver;
    marketingFeeReceiver = _marketingFeeReceiver;
}
```

Contract owner can change cooldown between trades settings

```
function cooldownEnabled(bool _status, uint8 _interval) public onlyOwner {
    buyCooldownEnabled = _status;
    cooldownTimerInterval = _interval;
}
```


Contract owner can change buyback settings

```
function setAutoBuybackSettings(bool _enabled, uint256 _cap, uint256 _amount, uint256 _period, bool
_autoBuybackMultiplier) external authorized {
    autoBuybackEnabled = _enabled;
    autoBuybackCap = _cap;
    autoBuybackAccumulator = 0;
    autoBuybackAmount = _amount;
    autoBuybackBlockPeriod = _period;
    autoBuybackBlockLast = block.number;
    autoBuybackMultiplier = _autoBuybackMultiplier;
}

function setBuybackMultiplierSettings(uint256 numerator, uint256 denominator, uint256 length) external
authorized {
    require(numerator / denominator <= 2 && numerator > denominator);
    buybackMultiplierNumerator = numerator;
    buybackMultiplierDenominator = denominator;
    buybackMultiplierLength = length;
}
```

Contract owner can transfer ownership

```
function transferOwnership(address payable adr) public onlyOwner {
    owner = adr;
    authorizations[adr] = true;
    emit OwnershipTransferred(adr);
}
```



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 no issue during the first review.

TOKEN DETAILS

Details

Buy fees:	2%
Sell fees:	18%
Max TX:	1,000,000,000,000
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



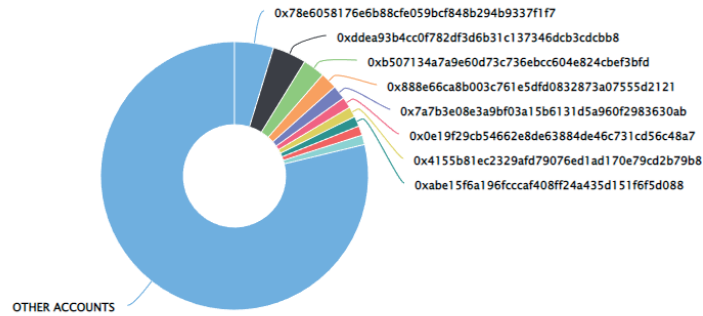
SUPERPANCAKE TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 21.25% (183,671,626,795.73 Tokens) of SuperPancake


Token Total Supply: 864,496,306,065.07 Token | Total Token Holders: 994

SuperPancake Top 10 Token Holders

Source: BscScan.com



(A total of 183,671,626,795.73 tokens held by the top 10 accounts from the total supply of 864,496,306,065.07 token)

Rank	Address	Quantity (Token)	Percentage
1	 0x78e6058176e6b88cfe059bcf848b294b9337f1f7	40,889,932,258.334817381345049626	4.7299%
2	0xddea93b4cc0f782df3d6b31c137346dc3cdcb8	35,216,872,957.43264428823168244	4.0737%
3	0xb507134a7a9e60d73c736ebcc604e824cbef3bfd	22,857,435,925.27988141384267764	2.6440%
4	0x888e66ca8b003c761e5dfd0832873a07555d2121	17,370,303,297.884396595312037356	2.0093%
5	0x7a7b3e08e3a9bf03a15b6131d5a960f2983630ab	14,597,062,932.367026828184180244	1.6885%
6	0x0e19f29cb54662e8de63884de46c731cd56c48a7	11,441,036,141.186285889018850877	1.3234%
7	0x4155b81ec2329afd79076ed1ad170e79cd2b79b8	10,875,764,555.932434541810301933	1.2580%
8	0xabe15f6a196fcca4f08ff24a435d151f6f5d088	10,623,218,727.316875933294624316	1.2288%
9	0xb42cfa26bc0cbe16a5146149f9a48105ba91dc49	9,900,000,000	1.1452%
10	0x7a2184937baa0f297b00ae684d92d66f138d6b32	9,900,000,000	1.1452%

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.

