freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

SOLT Gaming

$SOLT

14/02/2022

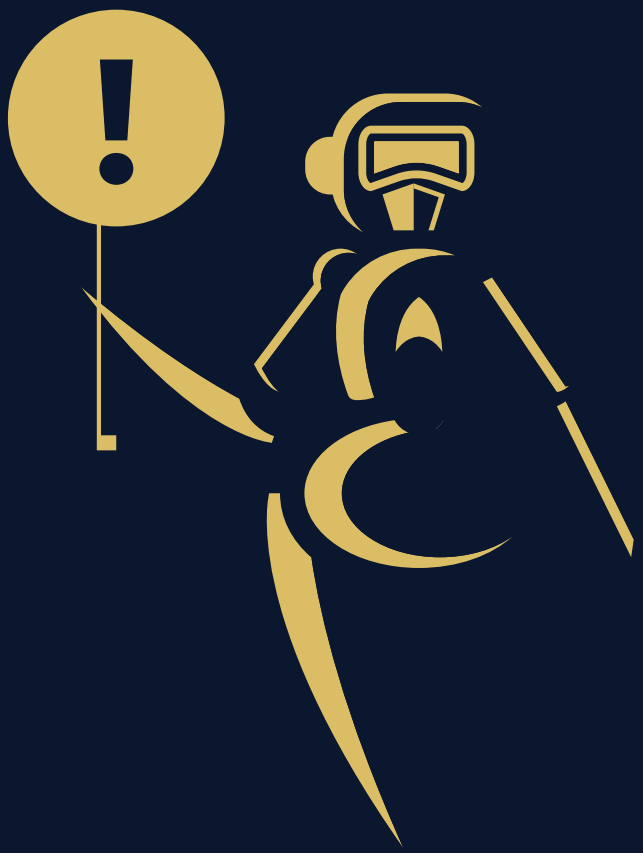
TABLE OF CONTENTS

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 SOLT Gaming (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0xb1d0929115889081ee9a274e7d995a8deafeb635

Network: Binance Smart Chain (BNB)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 14/02/2022

AUDIT OVERVIEW

97

Security Score

96

Static Scan
Automatic scanning for common vulnerabilities

99

ERC Scan
Automatic checks for ERC's conformance

0 High

0 Medium

0 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	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 exclude an address from transactions.

Contract owner can't mint tokens after initial contract deploy

Contract owner can exclude/include wallet from fees

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

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

Contract owner can exclude/include wallet from rewards

```
function excludeFromReward(address account) public onlyOwner() {
        require(!_isExcluded[account], "Account is already excluded");
        if(_rOwned[account] > 0) {
            _tOwned[account] = tokenFromReflection(_rOwned[account]);
        }
        _isExcluded[account] = true;
        _excluded.push(account);
}

function includeInReward(address account) external onlyOwner() {
        require(_isExcluded[account], "Account is already included");
        for (uint256 i = 0; i < _excluded.length; i++) {
            if (_excluded[i] == account) {
                _excluded[i] = _excluded[_excluded.length - 1];
                _tOwned[account] = 0;
                _isExcluded[account] = false;
                _excluded.pop();
                break;
            }
        }
}
```

Contract owner can change swap settings

```
function setSwapAndLiquifyEnabled(bool _enabled) public onlyOwner {
        swapAndLiquifyEnabled = _enabled;
        emit SwapAndLiquifyEnabledUpdated(_enabled);
}
```

Contract owner can change the fees up to 100%

```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
        _taxFee = taxFee;
}

function setDevelopmentFeePercent(uint256 developmentFee) external onlyOwner() {
        _developmentFee = developmentFee;
}

function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
        _liquidityFee = liquidityFee;
}
```

Contract owner can change max tx amount

```
function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner() {
        _maxTxAmount = _tTotal.mul(maxTxPercent).div(
            10**3
        );
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
        emit OwnershipTransferred(_owner, address(0));
        _owner = 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");
        emit OwnershipTransferred(_owner, newOwner);
        _owner = 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 no issue during the first review.

TOKEN DETAILS

Details

Buy fees:	N/A
Sell fees:	N/A
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

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

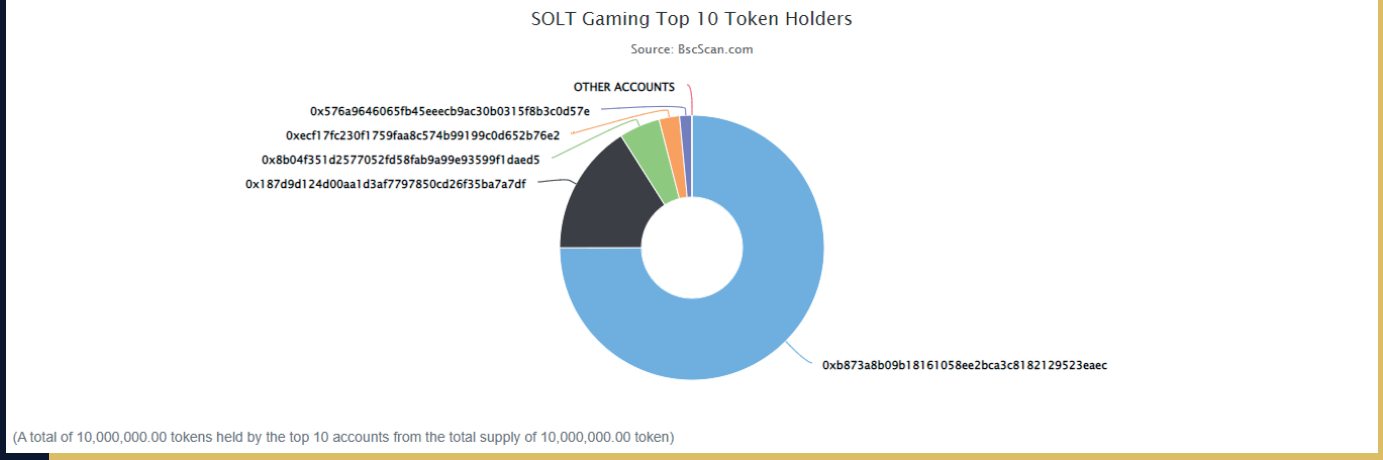
Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

SOLT GAMING TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (10,000,000.00 Tokens) of SOLT Gaming

Token Total Supply: 10,000,000.00 Token | Total Token Holders: 5



(A total of 10,000,000.00 tokens held by the top 10 accounts from the total supply of 10,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0xb873a8b09b18161058ee2bca3c8182129523eaec	7,498,400	74.9840%
2	0x187d9d124d00aa1d3af7797850cd26f35ba7a7df	1,601,600	16.0160%
3	0x8b04f351d2577052fd58fab9a99e93599f1daed5	500,000	5.0000%
4	0xecf17fc230f1759faa8c574b99199c0d652b76e2	250,000	2.5000%
5	0x576a9646065fb45eeecb9ac30b0315f8b3c0d57e	150,000	1.5000%

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.