freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

JEWEL VAULT

$JVT

30/08/2023

TOKEN OVERVIEW

Fees

- Buy fees: 12%
- Sell fees: 14%

Fees privileges

- Can’t change / set fees

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can’t change max tx amount and max wallet amount

Blacklist

- Blacklist function detected

Other privileges

- Can exclude / include from fees

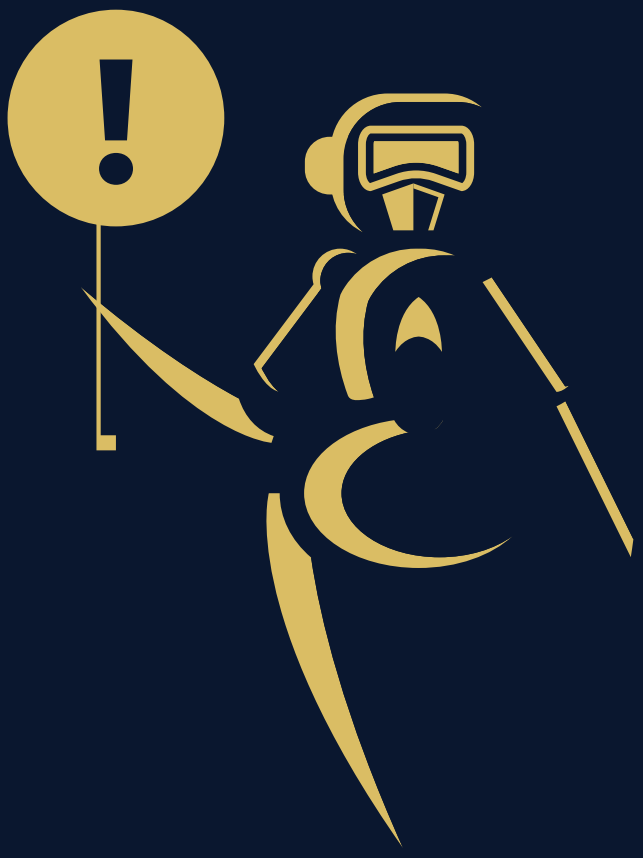
TABLE OF CONTENTS

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 JEWEL VAULT (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0x2C4A6C45f4fEef8E6e463Dd1C6D1479714b034DC

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 30/08/2023

WEBSITE DIAGNOSTIC

https://www.jewelvault.io

0-49 50-89 90-100

Performance Accessibility Best Practices SEO Progressive Web App

Socials

Twitter

https://twitter.com/JewelVaultToken

Telegram

https://t.me/JewelVaultToken

AUDIT OVERVIEW

88

Security Score

98

Static Scan
Automatic scanning for common vulnerabilities

84

ERC Scan
Automatic checks for ERC's conformance

1 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	Low
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 an address from transactions

```
function setBotBlacklist(address _botAddress, bool _flag)
    external
    onlyOwner
  {
    require(
      isContract(_botAddress),
      "only contract address, not allowed exteranlly owned account"
    );
    blacklist[_botAddress] = _flag;
}
```

- Contract owner can exclude wallet from tax

```
function setWhitelist(address _addr) external onlyOwner {
    _isFeeExempt[_addr] = true;
}
```

- Contract owner can change autoLiquidityReceiver, jewelvaultTreasuryReceiver, jewelInsuranceFundReceiver, blackHole, pairAddress and pairContract addresses

Current values:

autoLiquidityReceiver : 0xF1aE55a37Abd4ef37fF1540B559D29f4f2aDC460

jewelvaultTreasuryReceiver : 0x6C2EdC2924b21C9d2b063f5BEB6987C413d57bb4

jewelInsuranceFundReceiver : 0x5A431ceE7838291B863aa40D7017485Cd5E4A33e

blackHole : 0x23E111B9cc150F6Ee102Cba0451304FC331aE628

pairAddress : 0xcFDf34d474d96cE4b3B4a974413A96DBBC2DA18c

pairContract : 0xcFDf34d474d96cE4b3B4a974413A96DBBC2DA18c

```
function setFeeReceivers(
    address _autoLiquidityReceiver,
    address _jewelvaultTreasuryReceiver,
    address _jewelInsuranceFundReceiver,
    address _blackHole
  ) external onlyOwner {
    autoLiquidityReceiver = _autoLiquidityReceiver;
    jewelvaultTreasuryReceiver = _jewelvaultTreasuryReceiver;
    jewelInsuranceFundReceiver = _jewelInsuranceFundReceiver;
    blackHole = _blackHole;
}
```

```
function setPairAddress(address _pairAddress) external onlyOwner {
    pairAddress = _pairAddress;
}

function setLP(address _address) external onlyOwner {
    pairContract = IPancakeSwapPair(_address);
}
```

Contract owner can change rebase settings

Current values:

_autoRebase: false;

_isRebaseStarted: false;

_initRebaseStartTimeL: 1693296531 uint256

_lastRebasedTime: 1693296531 uint256

```
function setAutoRebase(bool _flag) external onlyOwner {
    if (_flag) {
        _autoRebase = _flag;
        _lastRebasedTime = block.timestamp;
    } else {
        _autoRebase = _flag;
    }
}

function startRebase() external onlyOwner {
    // execute only once
    require(!_isRebaseStarted, "Rebase already started");
    if (_isRebaseStarted) return;
    _initRebaseStartTime = block.timestamp;
    _lastRebasedTime = block.timestamp;
    _autoRebase = true;
    _isRebaseStarted = true;
}
```

Rebase settings:

```
...
if (deltaTimeFromInit < 365 days) {
        rebaseRate = 1818;
} else if (
        (deltaTimeFromInit >= 365 days) &&
        (deltaTimeFromInit < (15 * 365 days) / 10)
) {
        rebaseRate = 118;
} else if (
        (deltaTimeFromInit >= (15 * 365 days) / 10) &&
        (deltaTimeFromInit < (7 * 365 days))
) {
        rebaseRate = 9;
} else if (deltaTimeFromInit >= (7 * 365 days)) {
        rebaseRate = 2;
}
...
```

¬APY: 405,092.65% - duration > 1000 days

Contract owner can withdraw a certain token (Jewel token) from the contract's balance and swap it using a decentralized exchange router.

Native tokens not excluded

```
function withdrawAllToTreasury() external swapping onlyOwner {
    uint256 amountToSwap = _gonBalances[address(this)].div(
      _gonsPerFragment
    );
    require(
      amountToSwap > 0,
      "There is no Jewel Token token deposited in token contract"
    );
    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = router.WETH();
    router.swapExactTokensForETHSupportingFeeOnTransferTokens(
      amountToSwap,
      0,
      path,
      jewelvaultTreasuryReceiver,
      block.timestamp
    );
}
```

Contract owner can transfer ownership

```
function transferOwnership(address newOwner) public onlyOwner {
    _transferOwnership(newOwner);
}

function _transferOwnership(address newOwner) internal {
    require(newOwner != address(0));
    emit OwnershipTransferred(_owner, newOwner);
    _owner = newOwner;
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public onlyOwner {
    emit OwnershipRenounced(_owner);
    _owner = address(0);
}
```

Recommendation:

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. The risk can be prevented by temporarily locking the contract or renouncing ownership.

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 HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	12%
Sell fees:	14%
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

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

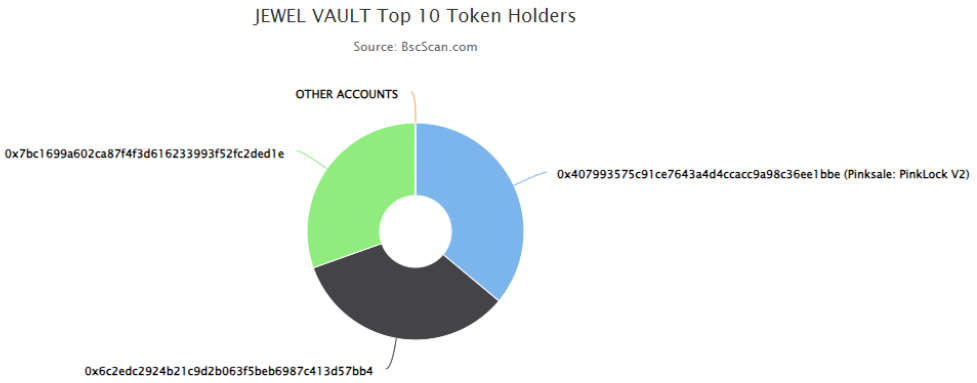
Rug Pull Risk

Liquidity:	N/A
Holders:	¬33% unlocked tokens

JVT TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (6,100,000.00 Tokens) of JEWEL VAULT

Token Total Supply: 6,100,000.00 Token | Total Token Holders: 3



(A total of 6,100,000.00 tokens held by the top 10 accounts from the total supply of 6,100,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	Pinksale: PinkLock V2	2,200,000	36.0656%
2	0x6C2EdC...13d57bb4	2,044,375	33.5143%
3	0x7Bc169...fc2deD1E	1,855,625	30.4201%

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.