freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

POV Chain
$POVChain

23/07/2023

TOKEN OVERVIEW

Fees

• **Buy fees:**   0%

• **Sell fees:**   3%

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 / or max wallet amount

Blacklist

• Blacklist function not detected

Other privileges

• Can exclude / include from fees

• Contract owner has to call enableTrading() function to enable trade

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
POV Chain (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0x29de166064E1f9467B70bd561e516564A5A9ba3c

Network: **Ethereum (ETH)**

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 23/07/2023

WEBSITE DIAGNOSTIC

https://povchain.org/

0-49　　　　**50-89**　　　　**90-100**

92
Performance

92
Accessibility

94
Best
Practices

91
SEO

NA
Progressive
Web App

Socials

Twitter

https://twitter.com/POVChain

Telegram

http://t.me/POVChainOfficial

AUDIT OVERVIEW

81

Security Score

98 Static Scan
Automatic scanning for common vulnerabilities

80 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	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 exclude/include wallet from tax**

```
function setNoFeeWallet(address account, bool enabled) public onlyOwner {
    _noFee[account] = enabled;
}
```

- **Contract owner has to call** enableTrading() **function to enable trade**

  Please note that any wallet excluded from fees/taxes can still engage in trading even if trading is disabled.

```
function enableTrading() external onlyOwner {
    require(!isTradingEnabled, "Trading already enabled");
    isTradingEnabled = true;
    emit _enableTrading();
}
…

if (isLimitedAddress(from,to)) {
    require(isTradingEnabled,"Trading is not enabled");
}
```

- **Contract owner can whitelist presale address**

```
function setPresaleAddress(address presale, bool yesno) external onlyOwner {
    require(isPresaleAddress[presale] != yesno,"Same bool");
    isPresaleAddress[presale] = yesno;
    _noFee[presale] = yesno;
    liquidityAdd[presale] = yesno;
    emit _setPresaleAddress(presale, yesno);
}
```

- **Contract owner can change** marketingAddress **and** marketing2Address **addresses**

  Default values:

  marketingAddress: 0x55628F08f4d947bBCd1c90509FeA9b71e2A099B0

  marketing2Address: 0x6bf35E66D227141616FDA303947861BA4348b2Bc

```
function changeWallets(address marketing, address marketing2) external onlyOwner {
    marketingAddress = payable(marketing);
    marketing2Address = payable(marketing2);
    emit _changeWallets(marketing, marketing2);
}
```

● Contract owner can change swap settings

```solidity
function toggleCanSwapFees(bool yesno) external onlyOwner {
    require(canSwapFees != yesno,"Bool is the same");
    canSwapFees = yesno;
    emit _toggleCanSwapFees(yesno);
}
```

● Contract owner can transfer ownership

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

function _setOwner(address newOwner) private {
    address oldOwner = _owner;
    _owner = newOwner;
    emit OwnershipTransferred(oldOwner, newOwner);
}
```

● Contract owner can renounce ownership

```solidity
function renounceOwnership() public virtual onlyOwner {
    _setOwner(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: 0%

Sell fees: 3%

Max TX: N/A

Max Sell: N/A

Honeypot Risk

Ownership: Owned

Blacklist: Not detected

Modify Max TX: Not detected

Modify Max Sell: Not detected

Disable Trading: Not detected

Rug Pull Risk

Liquidity: N/A

Holders: 100% unlocked tokens
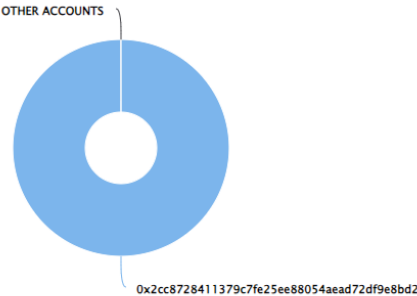
POVCHAIN TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (420,000,000,000,000,000.00 Tokens) of POV Chain   Token Total Supply: 420,000,000,000,000,000.00 Token | Total Token Holders: 1

POV Chain Top 10 Token Holders
Source: Etherscan.io

OTHER ACCOUNTS

0x2cc8728411379c7fe25ee88054aead72df9e8bd2

(A total of 420,000,000,000,000,000.00 tokens held by the top 10 accounts from the total supply of 420,000,000,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0x2cC872...df9e8Bd2	420,000,000,000,000,000	100.0000%

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.