Coinsult

Request your audit at coinsult.net

Advanced Manual
Smart Contract Audit

September 7, 2022

Audit requested by

Vital Veda

0x5c551B7f33e6f0A9C52eA99d37Cd409475b62C45 (testnet)

Table of Contents

Audit Summary

Audit Scope

Project Name	Vital Veda
Website	https://vitalveda.fit/
Blockchain	Binance Smart Chain
Smart Contract Language	Solidity
Contract Address	0x5c551B7f33e6f0A9C52eA99d37Cd409475b62C45 (testnet)
Audit Method	Static Analysis, Manual Review
Date of Audit	7 September 2022

This audit report has been prepared by Coinsult's experts at the request of the client. In this audit, the results of the static analysis and the manual code review will be presented. The purpose of the audit is to see if the functions work as intended, and to identify potential security issues within the smart contract.

The information in this report should be used to understand the risks associated with the smart contract. This report can be used as a guide for the development team on how the contract could possibly be improved by remediating the issues that were identified.

Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0xed47344473c9fe23b14f5f84306848263e2cfee4	8,800,000,000	50.0000%
2	0xd700ace74c6873c233d37729772ae3992e58819c	8,800,000,000	50.0000%

Source Code

Coinsult was comissioned by Vital Veda to perform an audit based on the following code:

https://testnet.bscscan.com/address/0x5c551B7f33e6f0A9C52eA99d37Cd409475b62C45#code

Disclaimer

This audit report has been prepared by Coinsult's experts at the request of the client. In this audit, the results of the static analysis and the manual code review will be presented. The purpose of the audit is to see if the functions work as intended, and to identify potential security issues within the smart contract.

The information in this report should be used to understand the risks associated with the smart contract. This report can be used as a guide for the development team on how the contract could possibly be improved by remediating the issues that were identified.

Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

Coinsult is not responsible for any financial losses. Nothing in this contract audit is financial advice, please do your own research.

The information provided in this audit is for informational purposes only and should not be considered investment advice. Coinsult does not endorse, recommend, support or suggest to invest in any project.

Coinsult can not be held responsible for when a project turns out to be a rug-pull, honeypot or scam.

Global Overview

Manual Code Review

In this audit report we will highlight the following issues:

Vulnerability Level	Total	Pending	Acknowledged	Resolved
Informational	0	0	0	0
Low-Risk	3	0	3	0
Medium-Risk	1	0	1	0
High-Risk	1	1	0	0

Privilege Overview

Coinsult checked the following privileges:

Contract Privilege	Description
Owner can mint?	Owner cannot mint new tokens
Owner can blacklist?	Owner can blacklist addresses
Owner can set fees > 25%?	Owner can set the sell fee to 25% or higher
Owner can exclude from fees?	Owner can exclude from fees
Owner can pause trading?	Owner can pause the smart contract
Owner can set Max TX amount?	Owner can set max transaction amount

More owner priviliges are listed later in the report.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Avoid relying on block.timestamp

block.timestamp can be manipulated by miners.

```
function permit(
    address owner,
    address spender,
    uint256 value,
    uint256 deadline,
    uint8 v,
    bytes32 r,
    bytes32 s
) public virtual override {
    require(block.timestamp &lt;= deadline, &quot;ERC20Permit: expired deadline&quot;);

    bytes32 structHash = keccak256(abi.encode(_PERMIT_TYPEHASH, owner, spender, value, _useNonce(owner)

    bytes32 hash = _hashTypedDataV4(structHash);

    address signer = ECDSA.recover(hash, v, r, s);
    require(signer == owner, &quot;ERC20Permit: invalid signature&quot;);

    _approve(owner, spender, value);
}
```

Recommendation

Do not use `block.timestamp`, `now` or `blockhash` as a source of randomness

Exploit scenario

```
contract Game {

    uint reward_determining_number;

    function guessing() external{
      reward_determining_number = uint256(block.blockhash(10000)) % 10;
    }
}
```

Eve is a miner. Eve calls `guessing` and re-orders the block containing the transaction. As a result, Eve wins the game.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Too many digits

Literals with many digits are difficult to read and review.

```
_mint(_msgSender(), 17600000000 * 10 ** decimals());
```

Recommendation

Use: Ether suffix, Time suffix, or The scientific notation

Exploit scenario

```
contract MyContract{
    uint 1_ether = 1000000000000000000;
}
```

While `1_ether` looks like `1 ether`, it is `10 ether`. As a result, it's likely to be used incorrectly.

🟢 **Low-Risk:** Could be fixed, will not bring problems.

Missing events arithmetic

Detect missing events for critical arithmetic parameters.

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

Recommendation

Emit an event for critical parameter changes.

Exploit scenario

```
contract C {

  modifier onlyAdmin {
    if (msg.sender != owner) throw;
    _;
  }

  function updateOwner(address newOwner) onlyAdmin external {
    owner = newOwner;
  }
}
```

`updateOwner()` has no event, so it is difficult to track off-chain changes in the buy price.

🟡 **Medium-Risk:** Should be fixed, could bring problems.

Duplicate usage of ‘maxTransferAmount’

```
uint256 maxTransferAmount = calculatePercent(
    percentOfTotalSupplyForAutoLiquidity,
    totalSupply()
);

uint256 contractTokenBalance = balanceOf(address(this));
bool overMinTokenBalance = contractTokenBalance &gt;= maxTransferAmount;
```

Recommendation

‘maxTransferAmount’ is used as a swap threshold, but also as a way to cap the transfer amount to a maximum. Use different parameters.

🔴 **High-Risk:** Must be fixed, will bring problems.

Owner can transfer funds from every blacklisted address to his own wallet

```
function transferFundsBack(
    address[] calldata _from,
    uint256[] calldata _amounts
) external onlyOwner {
    uint256 fromLength = _from.length;
    require(
        fromLength == _amounts.length,
        "Length of addresses and _amounts mismatch"
    );

    for (uint256 i = 0; i &lt; fromLength; ) {
        address from_ = _from[i];
        require(blacklist[from_], &quot;Address is not in the blacklist&quot;);

        _transfer(from_, owner(), _amounts[i]);

        unchecked {
            ++i;
        }
    }
}
```

Recommendation

Remove this function

Contract Privileges

Maximum Fee Limit Check

Coinsult tests if the owner of the smart contract can set the transfer, buy or sell fee to 25% or more. It is bad practice to set the fees to 25% or more, because owners can prevent healthy trading or even stop trading when the fees are set too high.

Type of fee	Description
Transfer fee	🔴 Owner can set the transfer fee to 25% or higher
Buy fee	🔴 Owner can set the buy fee to 25% or higher
Sell fee	🔴 Owner can set the sell fee to 25% or higher

Type of fee	Description
Max transfer fee	100%
Max buy fee	100%
Max sell fee	100%

Contract Pausability Check

Coinsult tests if the owner of the smart contract has the ability to pause the contract. If this is the case, users can no longer interact with the smart contract; users can no longer trade the token.

Privilege Check	Description
Can owner pause the contract?	🔴 Owner can pause the smart contract

Max Transaction Amount Check

Coinsult tests if the owner of the smart contract can set the maximum amount of a transaction. If the transaction exceeds this limit, the transaction will revert. Owners could prevent normal transactions to take place if they abuse this function.

Privilege Check	Description
Can owner set max tx amount?	🔴 Owner can set max transaction amount

Exclude From Fees Check

Coinsult tests if the owner of the smart contract can exclude addresses from paying tax fees. If the owner of the smart contract can exclude from fees, they could set high tax fees and exclude themselves from fees and benefit from 0% trading fees. However, some smart contracts require this function to exclude routers, dex, cex or other contracts / wallets from fees.

Privilege Check	Description
Can owner exclude from fees?	Owner can exclude from fees

Ability To Mint Check

Coinsult tests if the owner of the smart contract can mint new tokens. If the contract contains a mint function, we refer to the token's total supply as non-fixed, allowing the token owner to "mint" more tokens whenever they want.

A mint function in the smart contract allows minting tokens at a later stage. A method to disable minting can also be added to stop the minting process irreversibly.

Minting tokens is done by sending a transaction that creates new tokens inside of the token smart contract. With the help of the smart contract function, an unlimited number of tokens can be created without spending additional energy or money.

Privilege Check	Description
Can owner mint?	🟢 Owner cannot mint new tokens

Ability To Blacklist Check

Coinsult tests if the owner of the smart contract can blacklist accounts from interacting with the smart contract. Blacklisting methods allow the contract owner to enter wallet addresses which are not allowed to interact with the smart contract.

This method can be abused by token owners to prevent certain / all holders from trading the token. However, blacklists might be good for tokens that want to rule out certain addresses from interacting with a smart contract.

Privilege Check	Description
Can owner blacklist?	🔴 Owner can blacklist addresses

Other Owner Privileges Check

Coinsult lists all important contract methods which the owner can interact with.

✅ No other important owner privileges to mention.

Notes

Notes by Vital Veda

Notes: https://github.com/Coinsult/solidity/blob/main/notes_vitalveda.pdf

1- The owner can withdraw tokens from blacklisted addresses

Vitalveda Action: Code has been removed from the smart contract

2- Maximum Fee Limit Check
Coinsult tests if the owner of the smart contract can set the transfer, buy or sell fee to 25% or more. It is bad practice to set the fees to 25% or more, because owners can prevent healthy trading or even stop trading when the fees are set too high.

Vitalveda Action: Put a limit for a trading fee up to 25% max (% can be vary and adjust from 0 to 25%)

3- Max Transaction Amount Check
Coinsult tests if the owner of the smart contract can set the maximum amount of a transaction. If the transaction exceeds this limit; the transaction will revert. Owners could prevent normal transactions to take place if they abuse this function.

Vitalveda Action: we are putting a lower limit at 0.1%. Thus owner won't be able to set the max transaction limit lower than 0.1%.

Notes by Coinsult

✅ No notes provided by Coinsult

Contract Snapshot

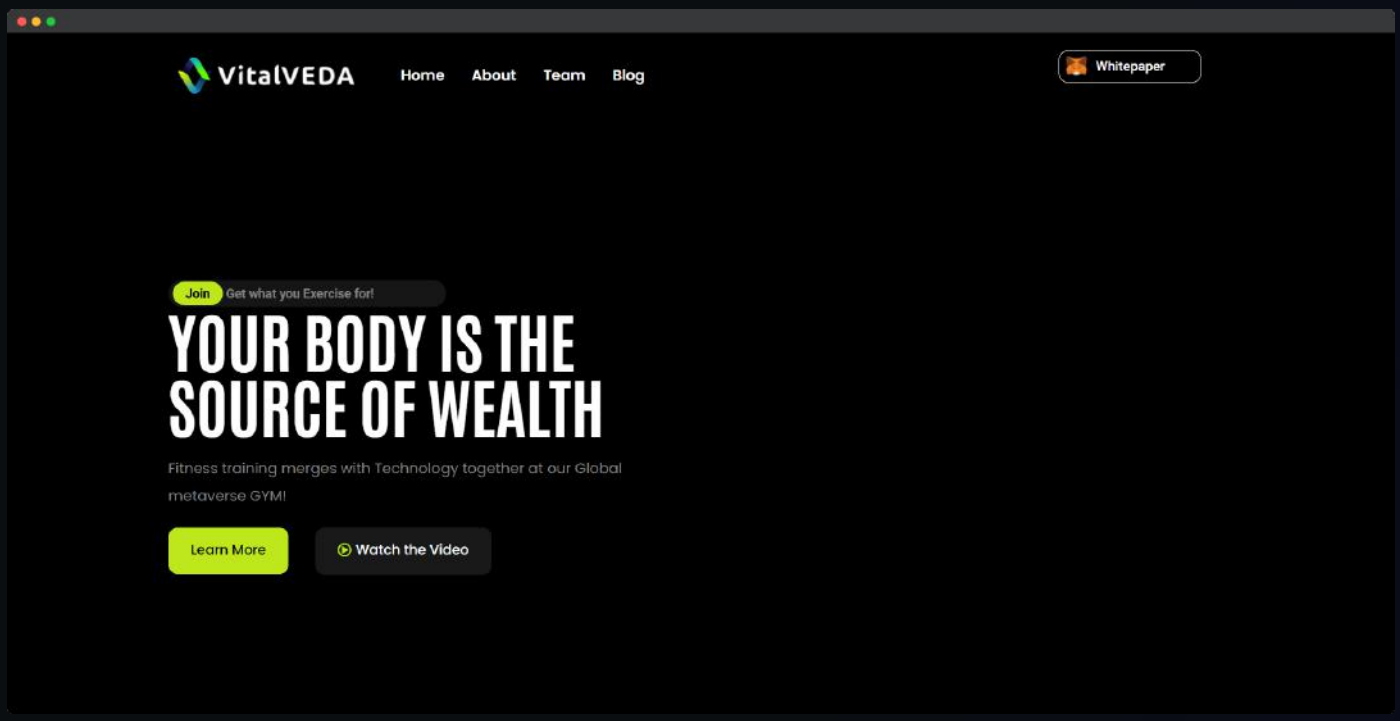
This is how the constructor of the contract looked at the time of auditing the smart contract.

```
contract VVFIT is ERC20, Pausable, Ownable, ERC20Permit {
//Divider which used in `calculatePercent` function
uint256 public PERCENT_DIVIDER_DECIMALS = 100000;

//Percent of tax taken on token sales
uint256 public salesTaxPercent;
```

Website Review

Coinsult checks the website completely manually and looks for visual, technical and textual errors. We also look at the security, speed and accessibility of the website. In short, a complete check to see if the website meets the current standard of the web development industry.



Type of check	Description
Mobile friendly?	The website is mobile friendly
Contains jQuery errors?	The website does not contain jQuery errors
Is SSL secured?	The website is SSL secured
Contains spelling errors?	The website does not contain spelling errors

Certificate of Proof

Not KYC verified by Coinsult

Vital Veda

Audited by Coinsult.net

Date: 7 September 2022

✔ Advanced Manual Smart Contract Audit

Coinsult

coinsult.net

End of report

Smart Contract Audit