BuildABlock Auditing Services
Class: Full Token Smart Contract

Basic Project Information

Project Name: Orbita
Contract Address:
0x903444e926F24E9C04416C46aAA7F6C4E8E4686a
Website: http://orbitadefi.com

Project Description:

An AI-powered Telegram management bot that can be a powerful tool for streamlining communication and managing messages more effectively.

Social Media Accounts:

TELEGRAM: **https://t.me/orbitadefilabportal**
TWITTER: https://twitter.com/Orbita_defilab

Date Of Audit: Thursday, June 8 2023

Overview of Set Smart Contract Tokenomics

Token Supply: 10,000,000 (10 Million)

Supply in PCS at launch: 8,650,000 (85.5%)

LIQUIDTY PERCENT LOCKED: 64.87%

LP LOCK VIA MUDRA:
https://mudra.website/?certificate=yes&type=0&lp=0x3eb9d70e312220a3bd00b34fa8737b0bc28b64a3

(Current Lock is subject to change)

Buy Fee: 8%
Sell Fee: 10%
Transfer fee: 10%
Max holdings per wallet: 2% of supply

Contract Call Functions : Owner

Mint: Impossible to mint new tokens

Fee: No Threshold on Fees (Can be set to 99%)

Max Tx: Max transaction can be set to 0%

Blacklist: Multi-Blacklist is available
(Note is primarily used for bot contracts and wallets via "AntiBot"& Bulk AntiBoy)

Trade: Trading cannot be disabled
(Trade can not be disabled via standard disable trade)

SECTION RESULT: FAILED

This section is considered a mild fail considering owner privileges can be used to conduct malicious activity.
However, despite malicious functions being available to the owner these are functions and privileges which a regularly seen on token smart contracts.

Users should exert caution when interacting with the smart contract, and ensure that they have done research on both the project & developer.

Authorized Privileges

```
function allowance(address owner, address spender) public view override returns (uint256) {
    return _allowances[owner][spender];
function approve(address spender, uint256 amount) public override returns (bool) {
    _approve(_msgSender(), spender, amount);
    return true;
  function transferFrom(address sender, address recipient, uint256 amount) public virtual override returns (bool) {
    transfer(sender, recipient, amount);
    uint256 currentAllowance = _allowances[sender][_msgSender()];
 function increaseAllowance(address spender, uint256 addedValue)
  function decreaseAllowance(address spender, uint256 subtractedValue)
 function isExcludedFromReward(address account) public view returns (bool) {
    return _isExcluded[account];
function excludeFromReward(address account) public onlyOwner()
  function includeInReward(address account) external onlyOwner()
  function excludeFromFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = true;
  function includeInFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = false;
  function isExcludedFromFee(address account) public view returns(bool)
  function setTaxes(uint256 _rfi, uint256 _marketing, uint256 _dev, uint256 _liquidity) public onlyOwner
  function setSellTaxes(uint256 _rfi, uint256 _marketing, uint256 _dev, uint256 _liquidity) public onlyOwner
  function _reflectRfi(uint256 rRfi, uint256 tRfi) private {
    _rTotal -=rRfi;
    totFeesPaid.rfi +=tRfi;
  function _takeLiquidity(uint256 rLiquidity, uint256 tLiquidity) private {
    totFeesPaid.liquidity +=tLiquidity;
  function _takeMarketing(uint256 rMarketing, uint256 tMarketing) private {
    totFeesPaid.marketing +=tMarketing;
  function _takeDev(uint256 rDev, uint256 tDev) private {
    totFeesPaid.dev += tDev;
function updateMarketingWallet(address newWallet) external onlyOwner{
    marketingAddress = newWallet;
  function updateDevWallet(address newDevWallet) external onlyOwner{
    devAddress = newDevWallet;
  function updateMaxWalletBalance(uint256 amount) external onlyOwner{
    maxWalletBalance = amount * 10**_decimals;
  function updatMaxBuyAmt(uint256 amount) external onlyOwner{
    maxBuyAmount = amount * 10**_decimals;
  function updatMaxSellAmt(uint256 amount) external onlyOwner{
    maxSellAmount = amount * 10**_decimals;
  function updateSwapTokensAtAmount(uint256 amount) external onlyOwner{
    swapTokensAtAmount = amount * 10**_decimals;
  function updateSwapEnabled(bool _enabled) external onlyOwner{
    swapEnabled = _enabled;
  function setAntibot(address account, bool state) external onlyOwner{
    require(_isBot[account] != state, 'Value already set');
    _isBot[account] = state;
  function bulkAntiBot(address[] memory accounts, bool state) external onlyOwner{
    for(uint256 i = 0; i < accounts.length; i++){
      _isBot[accounts[i]] = state;
  function updateRouterAndPair(address newRouter, address newPair) external onlyOwner{
    router = IRouter(newRouter);
    pair = newPair;
  function isBot(address account) public view returns(bool){
    return _isBot[account];
   function rescueBNB(uint256 weiAmount) external onlyOwner{
    require(address(this).balance >= weiAmount, "insufficient BNB balance");
    payable(msg.sender).transfer(weiAmount);
function rescueAnyBEP20Tokens(address _tokenAddr, address _to, uint _amount) public onlyOwner {
    require(_tokenAddr != address(this), "Cannot transfer out Token123!");
    IERC20(_tokenAddr).transfer(_to, _amount);
```

Link to SmartContract Code:

https://bscscan.com/address/0x903444e926f24e9c04416c46aaa7f6c4e8e4686a#code

General Issue Checking

N	Issue Description	Check Status
1	Compile Errors	PASS
2	Race Conditions and Reentrancy. Cross Function race conditions	PASS
3	Possible Delays in data delivery	PASS
4	Oracle Calls	PASS
5	Front Running + TimeStamp dependance	PASS
6	DoS block gas limit + DoS with Revert	PASS
7	Integer UnderFlow & OverFlow	UNCHECKED
8	Execution permissions + Economy Model	PASS
9	Design Logic + Safe Zeppelin module + Fallback function	PASS
10	Privacy user data leaks + Scooping & Declarations	UNCHECKED
11	Malicious Events Log	PASS

SECTION RESULT: PASSED

FULL AUDIT : PASSED

BUILD AUDIT DISCLAIMER

1

NOT RESPONSIBLE

BUILD A BLOCK holds NO responsibility for any and all actions done by the developer team of this audit's project. Build A Block holds no responsibility for the success, outcome and safety of the project itself (mentioned in the audit.)

2

NO GUARENTEE

Build A Block does not guarantee the safety of the team nor ensures the safety of the project. Information in this audit is all publicly viewable information that should be reviewed by the reader. An audit pass does NOT mean that the team and or project is fully safe, seeing as there are many ways of conducting illicit activities.

3

NOT FINANCIAL ADVICE

Build A Block is NOT a company or registered financial adviser. The audit should in no way influence anyone to purchase the mentioned project. ALL information should be independently verified and reviewed. When in doubt consult with a qualified and registered financial adviser. Build A Block holds no responsibility for any loss or suffering incurred by the mentioned project or this audit report.

DISCLAIMER

The information contained in this document is for INFORMATIVE purposes only and should not be considered financial advice. Information should be verified by the reader. Build A Block does not condone the project mentioned in this audit report.