Rodeo Solutions

Develop — Audit — Coach

$APP Token Smart Contract Audit

Rodeo Studios
July 2021

https://github.com/RodeoSolutions

Contents

Commission

Audited Project	**DAPPSY Token**
Project website	**Dappsy.io**
Contract Owner	**0x8c7D69Af1E419b6A75779FE10cF299eed737ac2f**
Smart Contract Address	**0x81e07CfF8a9331eF2A837B15a3560fb186bF5E8D**
Blockchain	**Binance Main Smart Chain**

Rodeo Solutions was commissioned by DAPPSY Token owners to perform an audit of their main smart contract. The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

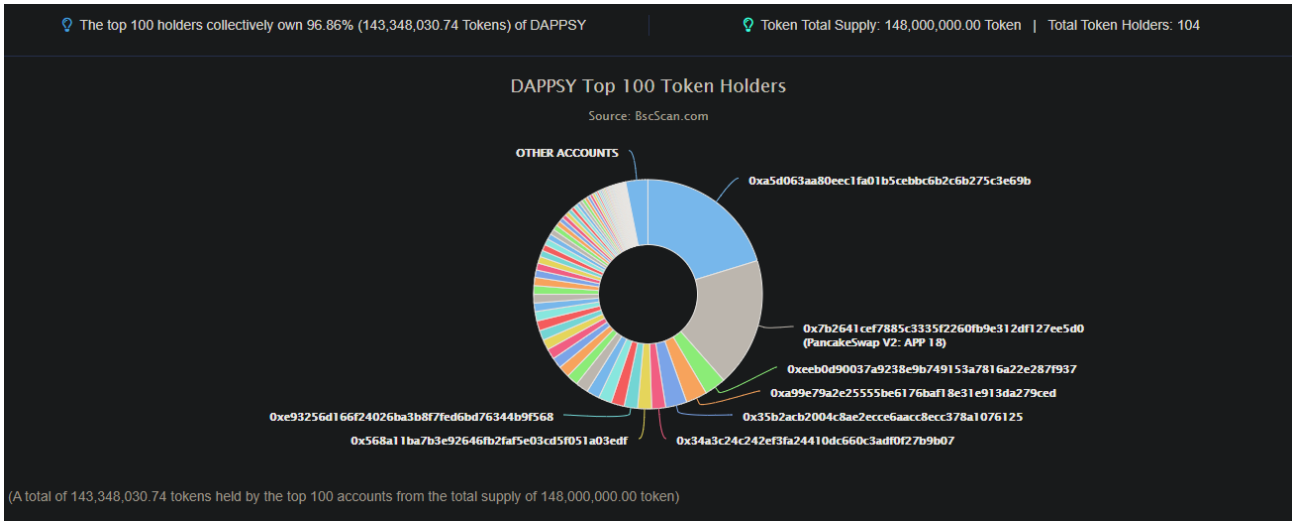
DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Rodeo Solutions and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Rodeo Solutions) owe no duty of care towards you or any other person, nor does Rodeo Solutions make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Rodeo Solutions hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Rodeo Solutions hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Rodeo Solutions, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

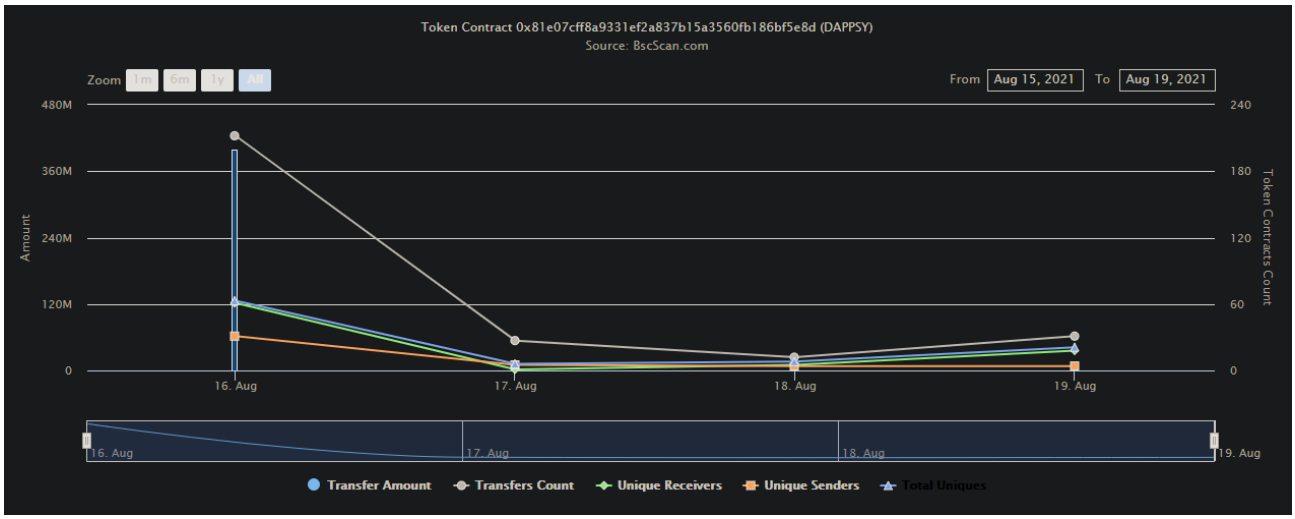
$APP Properties

Contract name	Token
Contract address	0x81e07CfF8a9331eF2A837B15a3560fb186bF5E8D
Total supply	148m
Token ticker	$APP
Decimals	8
Token holders	104
Transactions count	439
Top 100 holder's dominance	96.86%
Liquidity fee	5%
Tax fee	5%
Total fees	10%
Mintable	No
Burnable	Yes, manual
Uniswap V2 pair	0x7b2641cEF7885C3335F2260FB9e312df127ee5d0
Contract deployer address	0xd1E78C9D59746C4f9673038B45fAA999e586Ab75
Contract's current owner address	0x8c7D69Af1E419b6A75779FE10cF299eed737ac2f
Fee Address	0xA5d063AA80EEc1fA01b5cEBBC6B2C6B275C3e69b

As of 08/20/2021

Token Analytics





Contract Functions

Public

View

```
owner()
name()
symbol()
decimals()
totalSupply()
balanceOf(address account)
allowance(address owner, address spender)
isExcludedFromReward(address account)
totalFees()
reflectionFromToken(uint256 tAmount, bool deductTransferFee)
tokenFromReflection(uint256 rAmount)
buyBackUpperLimitAmount()
isExcludedFromFee(address account)
```

Virtual

```
increaseAllowance(address spender, uint256 addedValue)
decreaseAllowance(address spender, uint256 subtractedValue)
```

Executables

```
transfer(address recipient, uint256 amount)
approve(address spender, uint256 amount)
transferFrom(address sender, address recipient, uint256 amount)
deliver(uint256 tAmount)
```

Owner Executables

```
excludeFromReward(address account)
includeInReward(address account)
excludeFromFee(address account)
includeInFee(address account)
setAllFeePercent(uint8 taxFee, uint8 liquidityFee, uint8 burnFee, uint8 walletFee,
uint8 buybackFee)
setBuybackUpperLimit(uint256 buyBackLimit)
setMaxTxPercent(uint256 maxTxPercent)
setMaxWalletPercent(uint256 maxWalletPercent)
setSwapAndLiquifyEnabled(bool _enabled)
setFeeWallet(address payable newFeeWallet)
recoverBEP20(address tokenAddress, uint256 tokenAmount)
```

Checklist

Compiler errors.	Passed
Possible delays in data delivery.	Passed
Timestamp dependence.	Passed
Integer Overflow and Underflow.	Passed
DoS with Revert.	Passed
DoS with block gas limit.	Passed
Methods execution permissions.	Passed
Economy model of the contract.	Passed
Private user data leaks.	Passed
Malicious Events Log.	Passed
Scoping and Declarations.	Passed
Uninitialized storage pointers.	Passed
Arithmetic accuracy.	Passed
Design Logic.	Passed
Cross-function race conditions.	Passed
Fallback function security.	Passed
Safe Open Zeppelin contracts implementation and usage.	Passed
Whitepaper-Website-Contract correlation.	Low Severity

Owner privileges

DAPPSY Contract

- The owner can change fees at anytime

```
function setAllFeePercent(uint8 taxFee, uint8 liquidityFee, uint8 burnFee, uint8 walletFee, uint8 buybackFee) external onlyOwner() {
    require(taxFee >= 0 && taxFee <=maxTaxFee,"TF err");
    require(liquidityFee >= 0 && liquidityFee <=maxLiqFee,"LF err");
    require(burnFee >= 0 && burnFee <=maxBurnFee,"BF err");
    require(walletFee >= 0 && walletFee <=maxWalletFee,"WF err");
    require(buybackFee >= 0 && buybackFee <=maxBuybackFee,"BBF err");
    _taxFee = taxFee;
    _liquidityFee = liquidityFee;
    _burnFee = burnFee;
    _buybackFee = buybackFee;
    _walletFee = walletFee;
}
```

- The owner can exclude or include accounts from fees at anytime

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

```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Already excluded");
    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;
        }
    }
}
```

- The owner can exclude or include addresses from fees at any time

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

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

- The owner can set the MAX buyback amount

```
function setBuybackUpperLimit(uint256 buyBackLimit) external onlyOwner() {
    buyBackUpperLimit = buyBackLimit * 10**18;
}
```

- The owner can set the MAX transaction amount

```
function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner() {
    require(maxTxPercent >= minMxTxPercentage && maxTxPercent <=100,"err");
    _maxTxAmount = _tTotal.mul(maxTxPercent).div(
        10**2
    );
}
```

- The owner can set the MAX wallet holding percent

```
function setMaxWalletPercent(uint256 maxWalletPercent) external onlyOwner() {
    require(maxWalletPercent >= minMxWalletPercentage && maxWalletPercent <=100,"err");
    _maxWalletAmount = _tTotal.mul(maxWalletPercent).div(
        10**2
    );
}
```

- The owner can disable or enable Swap and Liquify

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

- The owner can change the Fee Wallet at any time

```
function setFeeWallet(address payable newFeeWallet) external onlyOwner {
    require(newFeeWallet != address(0), "ZERO ADDRESS");
    feeWallet = newFeeWallet;
}
```

- The owner can recover Tokens from other contract address

```
function recoverBEP20(address tokenAddress, uint256 tokenAmount) public onlyOwner {
    // do not allow recovering self token
    require(tokenAddress != address(this), "Self withdraw");
    IERC20(tokenAddress).transfer(owner(), tokenAmount);
}
```

Potential Issues

[Low] Whitepaper has outdated values

Currently the whitepaper states that the total supply is at 200m while the current contract total supply is 148m.

Another discrepancy is that the allocation would be among 5 wallets at the moment of deployment (Marketing, Reserve, Dev1, Dev2 and Contract Deployer) while at the moment the only two addresses are live. Contract deployer, holding the main flux of tokens, and a Fee Wallet, holding the fee amounts.

This does not present an issue to the Smart Contact itself.

Conclusions

The Whitepaper must be updated to reflect the contract's current state.

The Smart Contract code passed the audit successfully on the Binance Mainnet with no red flags or issues.