Cyberscope

Audit Report

ApeToday

April 2022

Type	BEP20
Network	BSC
Address	0x886fcb509b4ab3ccdf2f133cb89c5b4bd5c379f9
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	APTD
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0x886fcb509b4ab3ccdf2f133cb89c5b4bd5c379f9
Symbol	APTD
Decimals	9
Total Supply	70,000
Domain	apetoday.org

Source Files

Filename	**SHA256**
contract.sol	1087dd883ee4e44c997d79cacc2cc28971c8e38ddf9399585b4a78df3393f4bd

Audit Updates

Initial Audit	16th April 2022
Corrected	

Contract Analysis

🔴 Critical 🟠 Medium ⚪ Minor 🔵 Pass

Severity	Code	Description
🔴	ST	Contract Owner is not able to stop or pause transactions
🔵	OCTD	Contract Owner is not able to transfer tokens from specific address
🔵	OTUT	Owner Transfer User's Tokens
🔴	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
⚪	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
🔵	MT	Contract Owner is not able to mint new tokens
🔵	BT	Contract Owner is not able to burn tokens from specific wallet
🔵	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L751, 806

Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the _ tradingOpen to true and convert the contract into a honeypot.

```
if (!isAuthorized[sender]) {
            require(tradingOpen, "Trading not open yet");
        }
```

The owner can also convert it into a honeypot and prevent users from selling by setting the sellTotalFees to a very high value.

```
if (to == pair) {
            feeAmount = amount.mul(sellTotalFees).div(100);
        }
```

Recommendation

The contract could embody a check for not allowing setting the sellTotalFees less than a reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

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. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L831, 837

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the updateBuyFees and updateSellFees functions with a high percentage value.

```
function updateBuyFees(uint256 dev, uint256 liquidity) public onlyOwner {
        buyDevFee = dev;
        buyLiquidityFee = liquidity;
        buyTotalFees = dev.add(liquidity);
    }
```

```
function updateSellFees(uint256 dev, uint256 liquidity) public onlyOwner {
        selDevFee = dev;
        sellLiquidityFee = liquidity;
        sellTotalFees = dev.add(liquidity);
    }
```

Recommendation

The contract could embody a check for the maximum acceptable value.

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. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ULTW - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L826

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the clearStuckBalance function with any amountPercentage value.

```
function clearStuckBalance(uint256 amountPercentage) external onlyOwner {
        uint256 amountBNB = address(this).balance;
        payable(msg.sender).transfer((amountBNB * amountPercentage) / 100);
    }
```

```
uint256 marketingTokens =
contractTokenBalance.mul(marketingFee).div(totalFees);
swapAndSendToFee(marketingTokens);
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may violate the token's price.

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. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	FSA	Fixed Swap Address
●	CO	Code Optimization
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L05	Unused State Variable
●	L07	Missing Events Arithmetic

FSA - Fixed Swap Address

Criticality	minor
Location	contract.sol#L646

Description

The swap address is assigned once in the constructor and it can not be changed. The decentralized swaps sometimes create a new swap version or abandon the current. A contract that cannot change the swap address may not be able to catch-up the upgrade.

```
router = IUniswapV2Router02(0x10ED43C718714eb63d5aA57B78B54704E256024E);
        pair = IUniswapV2Factory(router.factory()).createPair(
            WBNB,
            address(this)
        );
```

Recommendation

It could be better to allow the swap address mutation in case of future swap updates.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L615, 619, 831, 836

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

```
// buy fees
    uint256 public buyDevFee = 2;
    uint256 public buyLiquidityFee = 3;
```

```
function updateBuyFees(uint256 dev, uint256 liquidity) public onlyOwner {
        buyDevFee = dev;
        buyLiquidityFee = liquidity;
        buyTotalFees = dev.add(liquidity);
    }
```

Recommendation

Remove buyDevFee, buyLiquidityFee, sellDevFee, sellLiquidityFee from code segments so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L559,567,670,674,678,682,831,837,843,853 and 1 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
whitelistPreSale
tradingStatus
updateSwapPercentages
updateSellFees
updateBuyFees
balanceOf
decimals
symbol
name
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L598,597,599,605

Description

Constant state variables should be declared constant to save gas.

```
_totalSupply
ZERO
WBNB
DEAD
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L25,853,857,946,950,597,598,599,601,602 and 5 more

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_allowances
_balances
_totalSupply
_decimals
_symbol
_name
ZERO
DEAD
WBNB
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L598,599

Description

There are segments that contain unused state variables.

```
ZERO
DEAD
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L831,837,843

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
liquiditySwap = liquidity
sellTotalFees = dev.add(liquidity)
buyTotalFees = dev.add(liquidity)
```

Recommendation

Emit an event for critical parameter changes.

Contract Functions

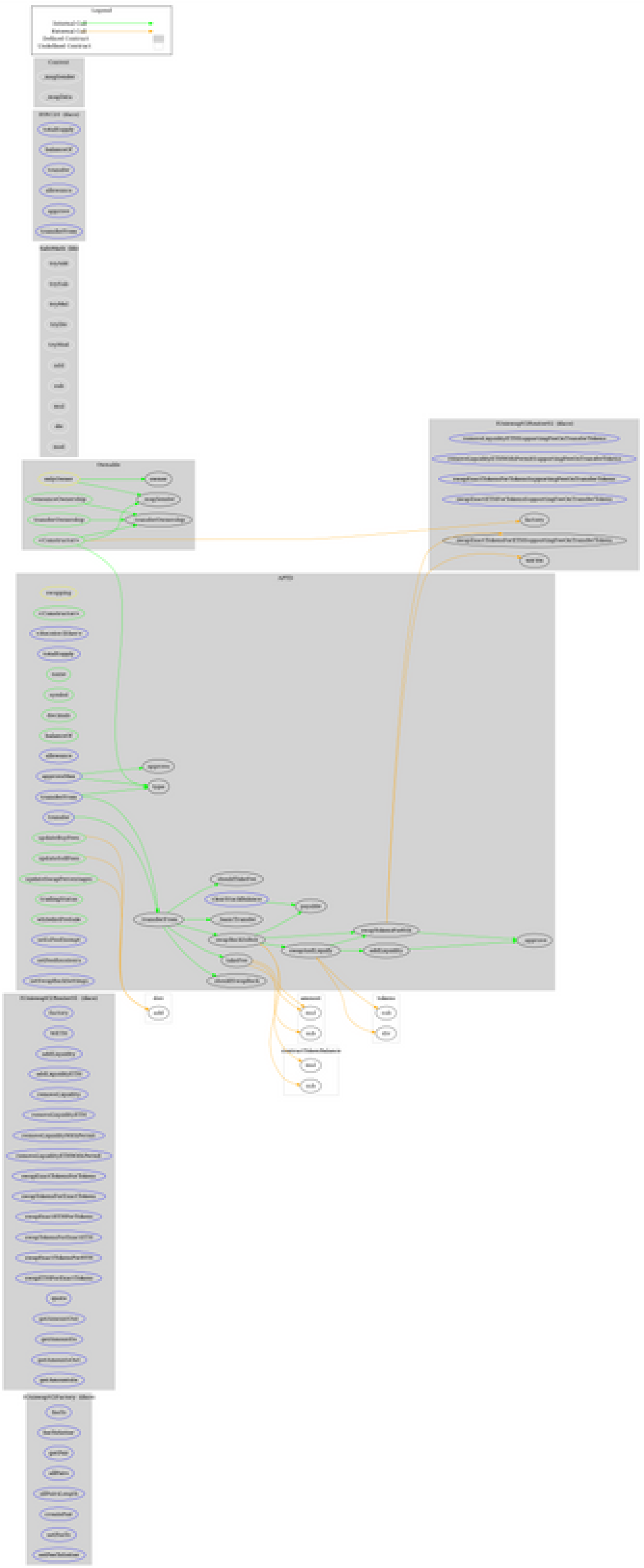
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-

	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-

	approve	External	✓	-
	transferFrom	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
APTD	Implementation	IERC20, Ownable		
	<Constructor>	Public	✓	-
	<Receive Ether>	External	Payable	-
	totalSupply	External		-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	balanceOf	Public		-
	allowance	External		-
	approve	Public	✓	-
	_approve	Internal	✓	
	approveMax	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	_transferFrom	Internal	✓	
	_basicTransfer	Internal	✓	
	shouldTakeFee	Internal		
	takeFee	Internal	✓	
	shouldSwapBack	Internal		
	clearStuckBalance	External	✓	onlyOwner

	updateBuyFees	Public	✓	onlyOwner
	updateSellFees	Public	✓	onlyOwner
	updateSwapPercentages	Public	✓	onlyOwner
	tradingStatus	Public	✓	onlyOwner
	whitelistPreSale	Public	✓	onlyOwner
	swapBackInBnb	Internal	✓	swapping
	swapAndLiquify	Private	✓	
	swapTokensForEth	Private	✓	
	addLiquidity	Private	✓	
	setIsFeeExempt	External	✓	onlyOwner
	setFeeReceivers	External	✓	onlyOwner
	setSwapBackSettings	External	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	apetoday.org
Registry Domain ID	D402200000019585179-LROR
Creation Date	2022-04-13T23:37:15Z
Updated Date	2022-04-15T10:44:37Z
Registry Expiry Date	2023-04-13T23:37:15Z
Registrar WHOIS Server	whois.tucows.com
Registrar URL	http://www.tucows.com
Registrar	Tucows Domains Inc.
Registrar IANA ID	69

The domain has been created 3 days before the creation of the audit. It will expire in 12 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

There are some functions that can be abused by the owner, like manipulating fees and stopping transactions. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions. The maximum fee percentage that can be set is 100%. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provides all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io