Cyberscope

Audit Report

Blockbusters Rewards Service

August 2022

SHA256 0ad528a23ecf3cf688050a6ca7745db772bff44d0f1c2ecded4caa8171ba60c8

Audited by © cyberscope

Table of Contents

Contract Review

Contract Name	BlockbustersRewardsService
Compiler Version	v0.8.15+commit.e14f2714
Testing Deploy	https://testnet.bscscan.com/token/0x59d5e91f09625ACE9891190F7aB6124A01B832F7
Domain	https://bbtftoken.com

Source Files

Filename	**SHA256**
contract.sol	0ad528a23ecf3cf688050a6ca7745db772bff44d0f1c2ecded4caa8171ba60c8

Audit Updates

Initial Audit	23rd August 2022
Corrected	

Introduction

The Blockbusters Rewards Service contract is responsible for handling the reward functionality. The main tasks are to keep the reward balance updated and distribute the rewards proportionally to the users.

Contract Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
Medium	ST	Stops Transactions	Unresolved
Minor	STC	Succeeded Transfer Check	Unresolved
Minor	MC	Missing Check	Unresolved
Minor	L01	Public Function could be Declared External	Unresolved
Minor	L02	State Variables could be Declared Constant	Unresolved
Minor	L04	Conformance to Solidity Naming Conventions	Unresolved
Minor	L05	Unused State Variable	Unresolved
Minor	L09	Dead Code Elimination	Unresolved
Minor	L12	Using Variables before Declaration	Unresolved
Minor	L13	Divide before Multiply Operation	Unresolved
Minor	L14	Uninitialized Variables in Local Scope	Unresolved
Minor	L15	Local Scope Variable Shadowing	Unresolved

ST - Stops Transactions

Criticality	medium
Location	contract.sol#L296
Status	Unresolved

Description

The contract is going to revert and stop the _process transaction on every _calculateFee, if the amount is lower than the fixed fee.

```
function _process(address from_, address to_, uint256 amount_) internal virtual returns (uint256){
        return _isServiceExempt(from_, to_) ? 0 : _calculateFee(from_, to_, amount_);
    }
```

Recommendation

A suggested implementation could apply the fees on each amount that is processed.

STC - Succeeded Transfer Check

Criticality	minor / informative
Location	contract.sol#L2187
Status	Unresolved

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
IERC20(reward_).transfer(account_, amount * _rewardDivisor);
```

Recommendation

The contract should check if the result of the transfer methods is successful.

MC - Missing Check

Criticality	minor / informative
Location	contract.sol#L2156
Status	Unresolved

Description

The contract is processing variables that have not properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues.

The method _withdraw should check if there is sufficient balance before every deposit.

```
function _withdraw(address to_) internal virtual {
        uint balance = address(this).balance;
        if (balance > 0) {
            address provider = _getProviderStorage();
            uint providerFee = _getProviderFeeStorage();
            if ( provider != address(0) && providerFee > 0) {
                _deposit(provider,  balance * _getProviderFeeStorage() / _PRECISION());
            }
            _deposit(to_, address(this).balance);
        }
    }
```

Prior to the swapExactETHForTokensSupportingFeeOnTransferTokens method the contract should check if there is sufficient balance before every swap.

```
function _swapForReward(address reward_, uint value_) internal {
        Snapshot storage rewardSnapshot = _snapshots[reward_];
        address[] storage path = _rewardPaths[reward_];
        uint value = value_ * _rewardRatio[reward_] / DEFAULT_PRECISION;
        uint oldRewardBalance = IERC20(reward_).balanceOf(address(this));

IUniswapV2Router02(_rewardRouters[reward_]).swapExactETHForTokensSupportingFeeOnTransf
erTokens{value: value}(0, path, address(this), block.timestamp + 1);
        _updateAccountSnapshot(rewardSnapshot, reward_,
_scaleAmount(IERC20(reward_).balanceOf(address(this)) - oldRewardBalance), 0);
    }
```

The method _process should check for reentrance.

```
function _process(address from_, address to_, uint amount_) internal override returns (uint256) {
        uint toFlags = _getFlags(to_);
        uint fromFlags = _getFlags(from_);
        if (!_isRewardSwapDisabled()) {
            uint balance = address(this).balance;
            if (balance > 0) {
                _swapForRewards(balance);
            }
        }
        uint128 amount = _scaleAmount(amount_);
        if (!_isRewardExempt(to_)) {
            if (!_isAccount(to_)) _addAccount(to_);
            _updateSnapshot(to_, amount, 0);
            _updateSnapshot(address(this), amount, 0);
        }
        if (!_isRewardExempt(from_)) {
            _updateSnapshot(from_, 0, amount);
            _updateSnapshot(address(this), 0, amount);
        }
        if (!_isRewardDistributionDisabled()) {
            _distribute(_isLPPair(from_), _isLPPair(to_));
        }
        return super._process(from_, to_, amount_);
    }
```

Recommendation

The contract should properly check the variables according to the required specifications.

L01 - Public Function could be Declared External

Criticality	minor / informative
Location	contract.sol#L1471,2025,1449,1478,1410,1442,1461,1278,1466,1284,1446,381,110
Status	Unresolved

Description

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

```
name
accounts
totalSupply
approve
transfer
allowance
symbol
nonces
decimals
...
```

Recommendation

Use the external attribute for functions never called from the contract.

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contract.sol#L1934
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
_reserved
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L165,1588,430,1926,15,312,410,153,1931,181,398,418,1564,1924,145,1560,1933,189,1930,394,406,1928,157,414,1590,173,338,402,1585,426,977,136,1925,169,970,1932,390,1884,149,1055,185,1725,1927,1124,161,422,321,1849,1929,193,1934,177
Status	Unresolved

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.

```
_SERVICE_EXEMPT_FLAG
_allowance
_REWARD_SWAP_DISABLED_FLAG
_rewardRouters
bits
_flags
_BLOCK_FROM_FLAG
_PROVIDER_FLAG
_distributionsPerBuy
...
```

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 / informative
Location	contract.sol#L1934,1055
Status	Unresolved

Description

There are segments that contain unused state variables.

```
_reserved
__gap
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor / informative
Location	contract.sol#L1983
Status	Unresolved

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.

```
(_distributionsPerBuy,_distributionsPerSell,_distributionsPerTransfer) =
(distributionsPerBuy_,distributionsPerSell_,distributionsPerTransfer_)
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor / informative
Location	contract.sol#L181,611,458,1592,450,1339,1289,1786,201,454,582,1524,1116,769,1535,592,80,924,557,917,64,1090,355,323,193,939,644,888,1566,1615,881,1760,1873,177,1271,1112,272,1320,1503,1485,1818,1855,185,760,1350,189,21,26,1782,898,654,60,197,232,2082,442,1886,434,1106,1102,625
Status	Unresolved

Description

Functions that are not used in the contract, and make the code's size bigger.

```
_ROUTER_FLAG
functionCallWithValue
_checkFlags
_initializeERC20WithStorage
_isTransferLimitExempt
_domainSeparator
_permit
currentAsAddress
_isServiceFeeExempt
...
```

Recommendation

Remove unused functions.

L12 - Using Variables before Declaration

Criticality	minor / informative
Location	contract.sol#L857
Status	Unresolved

Description

The contract is using a variable before the declaration. This is usually happening either if it has not been declared yet or the variable has been declared in a different scope.

```
slot
```

Recommendation

The variables should be declared before any usage of them.

L14 - Uninitialized Variables in Local Scope

Criticality	minor / informative
Location	contract.sol#L857,2183
Status	Unresolved

Description

The are variables that are defined in the local scope and are not initialized.

```
slot
amount
```

Recommendation

All the local scoped variables should be initialized.

L15 - Local Scope Variable Shadowing

Criticality	minor / informative
Location	contract.sol#L2026,287,286
Status	Unresolved

Description

The are variables that are defined in the local scope containing the same name from an upper scope.

```
accounts
providerFee
provider
```

Recommendation

The local variables should have different names from the upper scoped variables.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IService	Interface			
	process	External	✓	-
	withdraw	External	✓	-
	fee	External		-
	provider	External		-
	providerFee	External		-
bits	Library			
	only	Internal		
	all	Internal		
	any	Internal		
	check	Internal		
	all	Internal		
	set	Internal		
	toggle	Internal		
	isClear	Internal		
	clear	Internal		
	reset	Internal		
UsingFlags	Implementation			
	getFlags	Public		-
	_getFlags	Internal		
	_setFlags	Internal	✓	
	_getFlagStorage	Internal		
UsingPrecision	Implementation			
	_PRECISION	Internal		

UsingDefaultFlags	Implementation	UsingFlags		
	_INITIALIZED_FLAG	Internal		
	_TRANSFER_DISABLED_FLAG	Internal		
	_PROVIDER_FLAG	Internal		
	_SERVICE_FLAG	Internal		
	_NETWORK_FLAG	Internal		
	_SERVICE_EXEMPT_FLAG	Internal		
	_PROCESSING_FLAG	Internal		
	_ADMIN_FLAG	Internal		
	_BLOCKED_FLAG	Internal		
	_ROUTER_FLAG	Internal		
	_SERVICE_FEE_EXEMPT_FLAG	Internal		
	_SERVICES_DISABLED_FLAG	Internal		
	_FEE_EXEMPT_FLAG	Internal		
	_isFeeExempt	Internal		
	_isServiceFeeExempt	Internal		
	_isServiceExempt	Internal		
UsingAdmin	Implementation	UsingFlags, UsingDefaultFlags		
	_initializeAdmin	Internal	✓	
	setFlags	External	✓	requires
UsingFees	Implementation	UsingDefaultFlags, UsingPrecision		
	_setFee	Internal	✓	
	_getFee	Internal		
	_applyFee	Internal		
	_getFeesStorage	Internal		
UsingService	Implementation	IService, UsingAdmin, UsingFees		
	<Receive Ether>	External	Payable	-

	process	External	✓	requires
	withdraw	External	✓	requires
	provider	External		-
	providerFee	External		-
	fee	External		-
	_calculateFee	Internal		
	_deposit	Internal	✓	
	_withdraw	Internal	✓	
	_process	Internal	✓	
	_receive	Internal	✓	
	_getFeeStorage	Internal		
	_setFeeStorage	Internal	✓	
	_getProviderStorage	Internal		
	_getProviderFeeStorage	Internal		
UsingFlagsWithStorage	Implementation	UsingFlags		
	_getFlagStorage	Internal		
UsingFeesWithStorage	Implementation	UsingFees		
	_initializeFeesWithStorage	Internal	✓	
	_getFeesStorage	Internal		
UsingServiceWithStorage	Implementation	UsingService, UsingFeesWithStorage		
	_initializeServiceWithStorage	Internal	✓	
	_getProviderStorage	Internal		
	_getFeeStorage	Internal		
	_setFeeStorage	Internal	✓	
	_getProviderFeeStorage	Internal		
UsingInitializer	Implementation	UsingFlags, UsingDefaultFlags		

	initialized	Public		-
BlockbustersFlags	Implementation	UsingFlags, UsingDefaultFlags, UsingAdmin		
	_TRANSFER_LIMIT_DISABLED_FLAG	Internal		
	_LP_PAIR_FLAG	Internal		
	_REWARD_EXEMPT_FLAG	Internal		
	_TRANSFER_LIMIT_EXEMPT_FLAG	Internal		
	_ACCOUNT_FLAG	Internal		
	_BLOCK_FROM_FLAG	Internal		
	_BLOCK_TO_FLAG	Internal		
	_PER_TX_SELL_LIMIT_DISABLED_FLAG	Internal		
	_24HR_SELL_LIMIT_DISABLED_FLAG	Internal		
	_REWARD_DISTRIBUTION_DISABLED_FLAG	Internal		
	_REWARD_SWAP_DISABLED_FLAG	Internal		
	_isLPPair	Internal		
	_isLPPair	Internal		
	_isTransferLimitEnabled	Internal		
	_isRewardExempt	Internal		
	_isTransferLimitExempt	Internal		
	_isRouter	Internal		
	_checkFlags	Internal		
BlockbustersFlagsWithStorage	Implementation	UsingFlagsWithStorage, BlockbustersFlags		
IERC1822ProxiableUpgradeable	Interface			
	proxiableUUID	External		-

IBeaconUpgradeable	Interface			
	implementation	External		-
AddressUpgradeable	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	verifyCallResult	Internal		
StorageSlotUpgradeable	Library			
	getAddressSlot	Internal		
	getBooleanSlot	Internal		
	getBytes32Slot	Internal		
	getUint256Slot	Internal		
UsingERC1967UpgradeUpgradeable	Implementation			
	_getImplementation	Internal		
	_setImplementation	Private	✓	
	_upgradeTo	Internal	✓	
	_upgradeToAndCall	Internal	✓	
	_upgradeToAndCallUUPS	Internal	✓	
	_getAdmin	Internal		
	_setAdmin	Private	✓	
	_changeAdmin	Internal	✓	
	_getBeacon	Internal		
	_setBeacon	Private	✓	
	_upgradeBeaconToAndCall	Internal	✓	

	_functionDelegateCall	Private	✓	
UsingUUPS	Implementation	IERC1822ProxiableUpgradeable, UsingERC1967UpgradeUpgradeable		
	proxiableUUID	External		notDelegated
	upgradeTo	External	✓	onlyProxy
	upgradeToAndCall	External	Payable	onlyProxy
	_authorizeUpgrade	Internal	✓	
BlockbustersService	Implementation	UsingServiceWithStorage, UsingInitializer, BlockbustersFlagsWithStorage, UsingUUPS		
	_initializeBlockbustersService	Internal	✓	
	initialize	External	✓	initializer
	_authorizeUpgrade	Internal	✓	requires
Sets	Library			
	add	Internal	✓	
	remove	Internal	✓	
	get	Internal		
	pop	Internal	✓	
	contains	Internal		
	length	Internal		
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	✓	-
UsingPermit	Implementation			
	_initializePermits	Internal	✓	
	nonces	Public		-
	domainSeparators	Public		-
	_permit	Internal	✓	
	_updateDomainSeparator	Internal	✓	
	_domainSeparator	Private	✓	

	_recover	Internal		
	_getNameStorage	Internal		
	_getNoncesStorage	Internal		
	_getDomainSeparatorsStorage	Internal		
UsingERC20	Implementation	UsingPermit, UsingFlags, UsingDefaultFlags		
	transfer	Public	✓	requires
	transferFrom	External	✓	requires
	allowance	Public		-
	permit	Public	✓	-
	totalSupply	Public		-
	balanceOf	External		-
	symbol	Public		-
	decimals	Public		-
	name	Public		-
	approve	Public	✓	-
	_initializeERC20	Internal	✓	
	_allowanceFor	Internal		
	_approve	Internal	✓	
	_balanceOf	Internal		
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_getAllowanceStorage	Internal		
	_getBalancesStorage	Internal		
	_getTotalSupplyStorage	Internal		
	_setTotalSupplyStorage	Internal	✓	
	_getSymbolStorage	Internal		
	_getDecimalStorage	Internal		
UsingPermitWithStorage	Implementation	UsingPermit		
	_initializePermitWithStorage	Internal	✓	

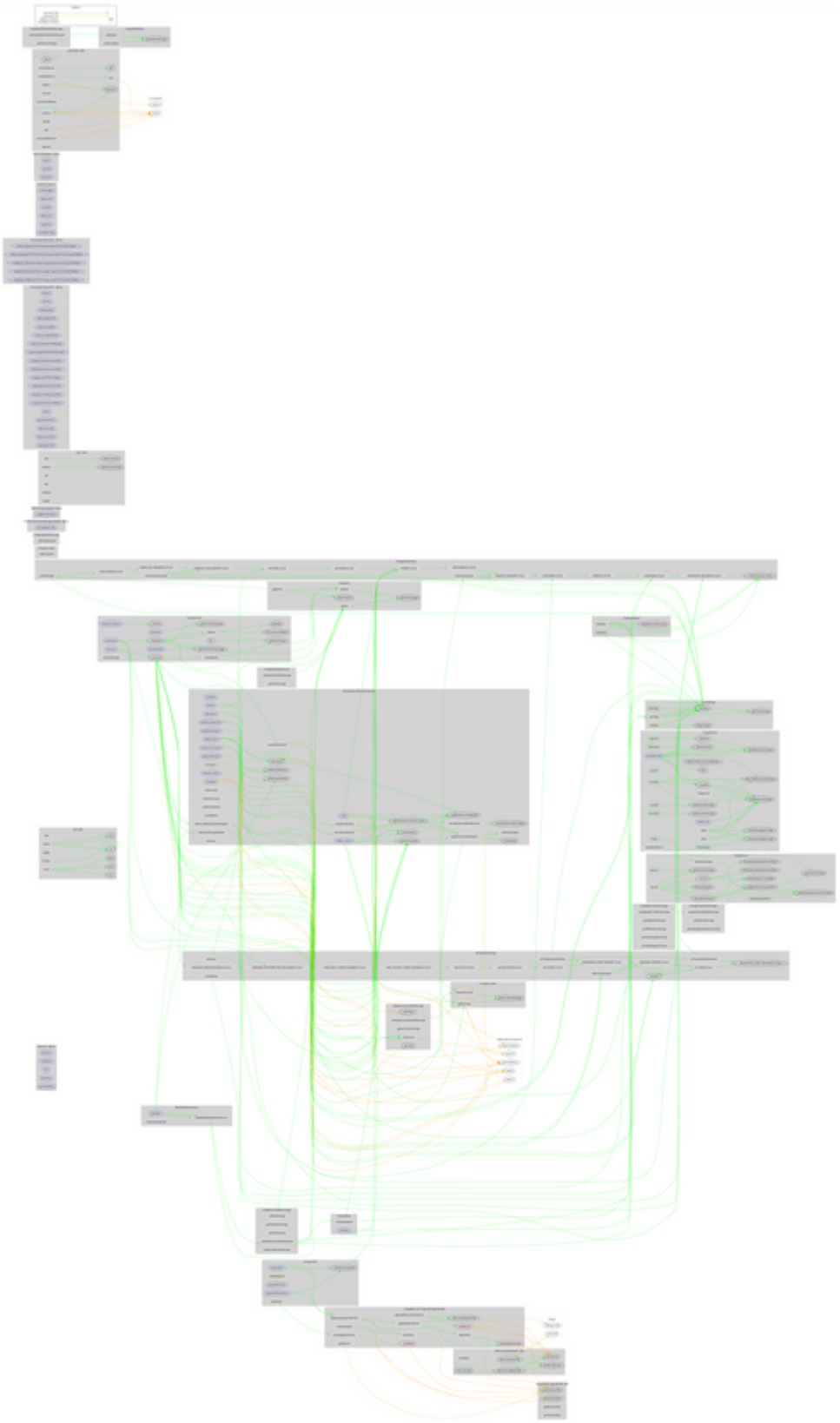
	_getNoncesStorage	Internal		
	_getDomainSeparatorsStorage	Internal		
UsingERC20WithStorage	Implementation	UsingERC20, UsingPermitWithStorage		
	_initializeERC20WithStorage	Internal	✓	
	_getBalancesStorage	Internal		
	_getAllowanceStorage	Internal		
	_getTotalSupplyStorage	Internal		
	_setTotalSupplyStorage	Internal	✓	
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
collections	Library			
	add	Internal	✓	
	add	Internal	✓	
	replace	Internal	✓	
	replace	Internal	✓	
	pop	Internal	✓	
	get	Internal		
	getAsAddress	Internal		
	next	Internal	✓	

	current	Internal		
	currentAsAddress	Internal		
	nextAsAddress	Internal	✓	
	length	Internal		
	remove	Internal	✓	
	remove	Internal	✓	
	itemsAsAddresses	Internal		
	indexOf	Internal		
UsingAccounts	Implementation			
	_addAccount	Internal	✓	
	_removeAccount	Internal	✓	
	_getAccountsStorage	Internal		
UsingAccountsWithStorage	Implementation	UsingAccounts		
	_isAccount	Internal	✓	
	_initializeAccountsWithStorage	Internal	✓	
	_getAccountsStorage	Internal		
UsingMultiToken	Implementation			
	_addToken	Internal	✓	
	_removeToken	Internal	✓	
	_getTokensStorage	Internal		
UsingMultiTokenWithStorage	Implementation	UsingMultiToken		
	_initializeMultiTokenWithStorage	Internal	✓	
	_getTokensStorage	Internal		

BlockbustersRewardsService	Implementation	BlockbustersService, UsingMultiTokenWithStorage, UsingAccountsWithStorage		
	initialize	External	✓	initializer
	setFee	External	✓	requires
	addAccounts	External	✓	requires
	removeAccounts	External	✓	requires
	setRewardRouter	External	✓	requires
	setDistributions	External	✓	requires
	replaceReward	External	✓	requires
	addReward	External	✓	requires
	claim	External	✓	requires
	distribute	External	✓	-
	accounts	Public		-
	totalAccounts	External		-
	snapshot	External		-
	_addAccount	Internal	✓	
	_removeAccount	Internal	✓	
	_addRewardToken	Internal	✓	
	_setRewardRouter	Internal	✓	
	_setRewardRatio	Internal	✓	
	_timestamp	Internal		
	_calculateUnclaimedRewards	Internal		
	_calculateRewardPerToken	Internal		
	_updateRewardSnapshot	Internal	✓	
	_isRewardToken	Internal	✓	
	_updateAccountSnapshot	Internal	✓	
	_updateSnapshot	Internal	✓	
	_scaleAmount	Internal		
	_calculateFee	Internal		
	_swapForReward	Internal	✓	
	_swapForRewards	Internal	✓	

	_isRewardDistributionDisabled	Internal		
	_isRewardSwapDisabled	Internal		
	_process	Internal	✓	
	_distributeRewardToAccount	Internal	✓	
	_distributeReward	Internal	✓	
	_distribute	Internal	✓	
	_distribute	Internal	✓	

Contract Flow



Domain Info

Domain Name	bbtftoken.com
Registry Domain ID	2685924176_DOMAIN_COM-VRSN
Creation Date	2022-03-31T18:04:42Z
Updated Date	2022-03-31T18:04:43Z
Registry Expiry Date	2023-03-31T18:04:42Z
Registrar WHOIS Server	whois.godaddy.com
Registrar URL	https://www.godaddy.com
Registrar	GoDaddy.com, LLC
Registrar IANA ID	146

The domain was created 5 months before the creation of the audit. It will expire in 7 months.

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

Summary

This audit focuses on the business logic issues, the security concerns and the potential improvements. The owner has the authority to upgrade the contract via proxy.

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 provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io