



Cyberscope

Audit Report

Baltic Miners

July 2022

Type BEP20

Network BSC

Address 0xe247d974a7AdCBc097764c6d76C164211e50b3e0

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Contract Analysis	4
ST - Stop Transactions	5
Description	5
Recommendation	5
ULTW - Unlimited Liquidity to Team Wallet	6
Description	6
Recommendation	6
BC - Blacklisted Contracts	7
Description	7
Recommendation	7
Contract Diagnostics	8
US - Untrusted Source	9
Description	9
Recommendation	9
STC - Succeeded Transfer Check	10
Description	10
Recommendation	10
MTS - Manipulate Total Supply	11
Description	11
Recommendation	11
L01 - Public Function could be Declared External	12
Description	12

Recommendation	12
L02 - State Variables could be Declared Constant	13
Description	13
Recommendation	13
L04 - Conformance to Solidity Naming Conventions	14
Description	14
Recommendation	14
L05 - Unused State Variable	15
Description	15
Recommendation	15
L07 - Missing Events Arithmetic	16
Description	16
Recommendation	16
L12 - Using Variables before Declaration	17
Description	17
Recommendation	17
L13 - Divide before Multiply Operation	18
Description	18
Recommendation	18
L14 - Uninitialized Variables in Local Scope	19
Description	19
Recommendation	19
Contract Functions	20
Contract Flow	24
Domain Info	25
Summary	26
Disclaimer	27
About Cyberscope	28

Contract Review

Contract Name	BalticFinancialToken
Compiler Version	v0.8.15+commit.e14f2714
Optimization	5000 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xe247d974a7AdCBc097764c6d76C164211e50b3e0
Symbol	BMFT
Decimals	5
Total Supply	4,000,000,000
Domain	http://www.balticminers.com/

Source Files

Filename	SHA256
contract.sol	bae3013b5f5ca523ec5894012f117e71e6f5cde41153f70527f6f403ed4cf221

Audit Updates

Initial Audit	7th July 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	medium
Location	contract.sol#L558

Description

The contract owner has the authority to stop the sales for specific users excluding the owner. The owner may take advantage of it by setting the `limitPercent` to zero. As a result, the `sellLimitPerTime` will be zero and the expression will always revert.

```
require(amount <= userLimits[from].sellLimitPerTime, "Limited wallet selling above limit.");
```

Recommendation

The contract could embody a check for not allowing setting the `limitPercent` 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.

ULTW - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L676

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 `sweepContingency` method.

```
function sweepContingency() external onlyOwner {  
    require(!_hasLiqBeenAdded, "Cannot call after liquidity.");  
    payable(_owner).transfer(address(this).balance);  
}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile 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.

BC - Blacklisted Contracts

Criticality	medium
Location	contract.sol#L697

Description

The contract owner has the authority to stop contracts from transactions. The owner may take advantage of it by calling the `blacklistAddress` function.

```
try antiSnipe.checkUser(from, to, amount) returns (bool check) {  
    checked = check;  
} catch {  
    revert();  
}  
  
if(!checked) {  
    revert();  
}
```

Recommendation

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
●	US	Untrusted Source
●	STC	Succeeded Transfer Check
●	MTS	Manipulate Total Supply
●	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
●	L12	Using Variables before Declaration
●	L13	Divide before Multiply Operation
●	L14	Uninitialized Variables in Local Scope

US - Untrusted Source

Criticality	medium
Location	contract.sol#L697

Description

The contract uses an external contract in order to determine the transaction's flow. The external contract is untrusted. As a result it may produce security issues and harm the transactions.

```
try antiSnipe.checkUser(from, to, amount) returns (bool check) {
```

Recommendation

The contract should use a trusted external source. A trusted source could be either a commonly recognized or an audited contract. The pointing addresses should not be able to change after the initialization.

STC - Succeeded Transfer Check

Criticality

minor

Location

contract.sol#L676

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.

```
function sweepContingency() external onlyOwner {  
    require(!_hasLiqBeenAdded, "Cannot call after liquidity.");  
    payable(_owner).transfer(address(this).balance);  
}
```

Recommendation

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

MTS - Manipulate Total Supply

Criticality	minor
Location	contract.sol#L580

Description

Owner is able to manipulate total supply. This change will have a direct impact on the token price and Market Cap.

```
uint256 rebaseMinutes = rebaseTimeInMinutes * 1 minutes;
if(autoRebaseEnabled
    && block.timestamp >= autoRebaseLastTriggered + rebaseMinutes
){
    uint256 deltaTime = block.timestamp - autoRebaseLastTriggered;
    uint256 rebaseAmounts = deltaTime / (rebaseMinutes);
    uint256 epoch = rebaseAmounts * rebaseTimeInMinutes;

    for (uint256 i = 0; i < rebaseAmounts; i++) {
        _tTotal = (_tTotal * ((10**_rateDecimals) + rebaseRate)) / (10**_rateDecimals);
    }
}
```

Recommendation

The contract owner should carefully manage the adjustment of the circulating supply (increases or decreases), according to the token's price fluctuations.

L01 - Public Function could be Declared External

Criticality

minor

Location

contract.sol#L658,384,315,320,380

Description

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

```
isExcludedFromLimits  
approve  
transfer  
isExcludedFromFees  
enableTrading
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality

minor

Location

contract.sol#L192,191

Description

Constant state variables should be declared constant to save gas.

```
_maxTxAmount  
_maxWalletSize
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality

minor

Location

contract.sol#L161,33,165,133,141,160,117,114,157,159,149,417,158,116,177,118,195,203

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.

```
_rateDecimals  
_hasLiqBeenAdded  
_decimals  
_taxWallets  
_name  
_antiSnipe  
maxSellTaxes  
_antiBlock  
_transferTaxes  
...
```

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#L192,191

Description

There are segments that contain unused state variables.

```
_maxTxAmount  
_maxWalletSize
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality

minor

Location

contract.sol#L463,457,500,484

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.

```
rebaseRate = rate
limitTime = timeInMinutes * 60
swapThreshold = (_tTotal * thresholdPercent) / thresholdDivisor
piSwapPercent = priceImpactSwapPercent
```

Recommendation

Emit an event for critical parameter changes.

L12 - Using Variables before Declaration

Criticality

minor

Location

contract.sol#L693

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.

check

Recommendation

The variables should be declared before any usage of them.

L13 - Divide before Multiply Operation

Criticality

minor

Location

contract.sol#L726,532

Description

Performing divisions before multiplications may cause lose of prediction.

```
treasuryAmount = (feeAmount / currentFee.total) * currentFee.treasury  
insuranceAmount = (feeAmount / currentFee.total) * currentFee.insurance  
firePitAmount = (feeAmount / currentFee.total) * currentFee.firePit  
rebaseAmounts = deltaTime / (rebaseMinutes)  
feeAmount = (amount / masterTaxDivisor) * currentFee.total
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality

minor

Location

contract.sol#L693,692

Description

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

checked
check

Recommendation

All the local scoped variables should be initialized.

Contract Functions

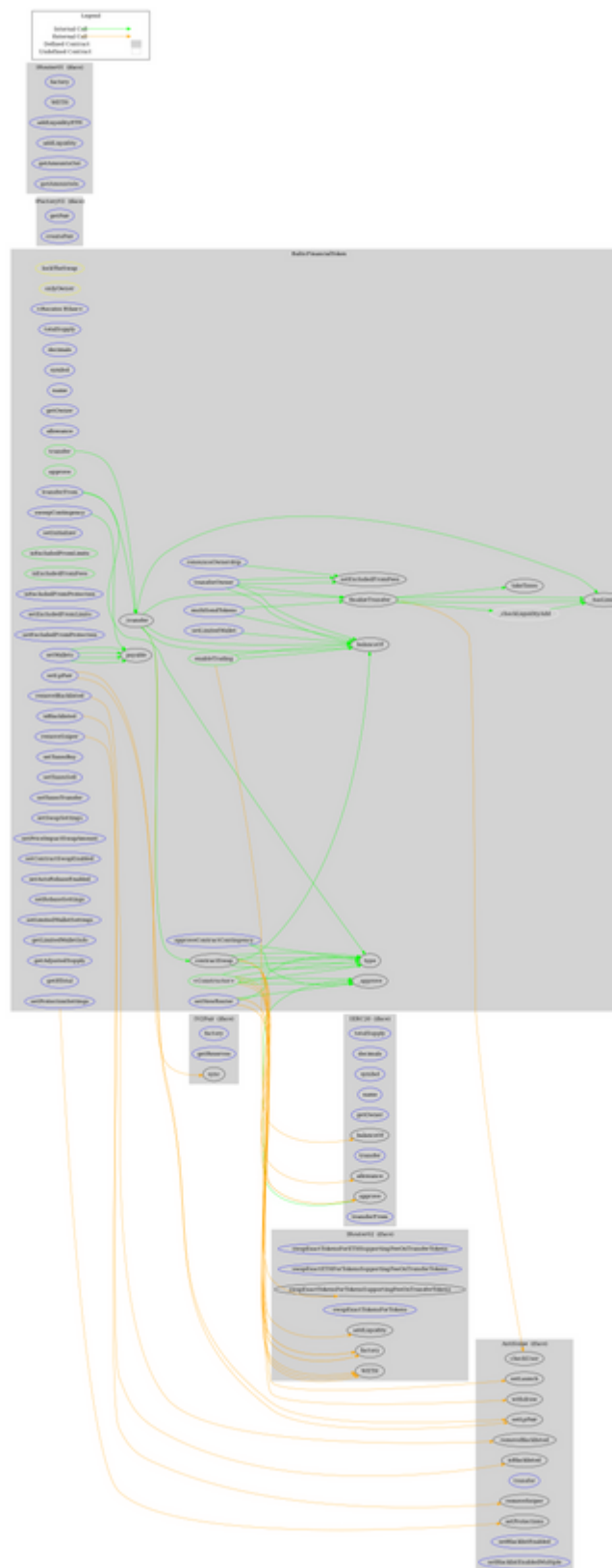
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IFactoryV2	Interface			
	getPair	External		-
	createPair	External	✓	-
IV2Pair	Interface			
	factory	External		-
	getReserves	External		-
	sync	External	✓	-
IRouter01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
	addLiquidity	External	✓	-
	getAmountsOut	External		-
	getAmountsIn	External		-

IRouter02	Interface	IRouter01		
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokens	External	✓	-
AntiSnipe	Interface			
	checkUser	External	✓	-
	setLaunch	External	✓	-
	setLpPair	External	✓	-
	setProtections	External	✓	-
	removeSniper	External	✓	-
	removeBlacklisted	External	✓	-
	transfer	External	✓	-
	withdraw	External	✓	-
	isBlacklisted	External		-
	setBlacklistEnabled	External	✓	-
	setBlacklistEnabledMultiple	External	✓	-
BalticFinancialToken	Implementation	IERC20		
	<Constructor>	Public	Payable	-
	<Receive Ether>	External	Payable	-
	transferOwner	External	✓	onlyOwner
	renounceOwnership	External	✓	onlyOwner
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	allowance	External		-
	balanceOf	Public		-

	transfer	Public	✓	-
	approve	Public	✓	-
	_approve	Internal	✓	
	approveContractContingency	External	✓	onlyOwner
	transferFrom	External	✓	-
	setNewRouter	External	✓	onlyOwner
	setLpPair	External	✓	onlyOwner
	setInitializer	External	✓	onlyOwner
	isExcludedFromLimits	Public		-
	isExcludedFromFees	Public		-
	isExcludedFromProtection	External		-
	setExcludedFromFees	Public	✓	onlyOwner
	setExcludedFromLimits	External	✓	onlyOwner
	setExcludedFromProtection	External	✓	onlyOwner
	removeBlacklisted	External	✓	onlyOwner
	isBlacklisted	External		-
	removeSniper	External	✓	onlyOwner
	setProtectionSettings	External	✓	onlyOwner
	setWallets	External	✓	onlyOwner
	setTaxesBuy	External	✓	onlyOwner
	setTaxesSell	External	✓	onlyOwner
	setTaxesTransfer	External	✓	onlyOwner
	setSwapSettings	External	✓	onlyOwner
	setPriceImpactSwapAmount	External	✓	onlyOwner
	setContractSwapEnabled	External	✓	onlyOwner
	setAutoRebaseEnabled	External	✓	onlyOwner
	setRebaseSettings	External	✓	onlyOwner
	setLimitedWallet	External	✓	onlyOwner
	setLimitedWalletSettings	External	✓	onlyOwner
	getLimitedWalletInfo	External		-
	getAdjustedSupply	External		-
	getRTotal	External		-
	_hasLimits	Internal		
	_transfer	Internal	✓	

	contractSwap	Internal	✓	lockTheSwap
	_checkLiquidityAdd	Internal	✓	
	enableTrading	Public	✓	onlyOwner
	sweepContingency	External	✓	onlyOwner
	multiSendTokens	External	✓	onlyOwner
	finalizeTransfer	Internal	✓	
	takeTaxes	Internal	✓	

Contract Flow



Domain Info

Domain Name	balticminers.com
Registry Domain ID	2676005082_DOMAIN_COM-VRSN
Creation Date	2022-02-18T13:35:33Z
Updated Date	2022-02-18T13:35:33Z
Registry Expiry Date	2023-02-18T13:35:33Z
Registrar WHOIS Server	whois.1api.net
Registrar URL	http://www.1api.net
Registrar	1API GmbH
Registrar IANA ID	1387

The domain has been created in 8 months before the creation of the audit.

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 transferring funds to the team's wallet, blacklisting addresses and preventing specific users from selling. In addition, an untrusted source is used for core functionalities on the contract implementation. 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 provide all the essential tools to assist users draw their own conclusions.



The Cyberscope team

<https://www.cyberscope.io>



Cyberscope

Audit Report

Baltic Miners Staking

May 2022

Type BEP20

Network BSC

Address 0x10008DB953Db6bD170A22bA135883a3cCB1bD470

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Contract Diagnostics	4
STC - Succeeded Transfer Check	5
Description	5
Recommendation	5
CO - Code Optimization	6
Description	6
Recommendation	6
DSRC - Data Structure Range Check	7
Description	7
Recommendation	7
BA - Blacklist Addresses	8
Description	8
Recommendation	8
TAM - Token Address Mutation	9
Description	9
Recommendation	9
FO - Fees Overflow	10
Description	10
Recommendation	10
OCTD - Owner Contract Tokens Drain	12
Description	12
Recommendation	12

L01 - Public Function could be Declared External	13
Description	13
Recommendation	13
L04 - Conformance to Solidity Naming Conventions	14
Description	14
Recommendation	14
L09 - Dead Code Elimination	15
Description	15
Recommendation	15
Contract Functions	16
Contract Flow	20
Summary	21
Disclaimer	22
About Cyberscope	23

Contract Review

Contract Name	BMIStaking
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	Unlicense
Explorer	https://bscscan.com/token/0x10008DB953Db6bD170A22bA135883a3cCB1bD470

Source Files

Filename	SHA256
contract.sol	b137615bc3d28fb39da651da676175ac0ec8738db952eb616681fc1aa2545411

Audit Updates

Initial Audit	21st June 2022
Corrected	

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description
●	STC	Succeeded Transfer Check
●	CO	Code Optimization
●	DSRC	DSRC
●	BA	Blacklist Addresses
●	TAM	Token Address Mutation
●	FO	Fees Overflow
●	OCTD	Owner Contract Tokens Drain
●	L01	Public Function could be Declared External
●	L04	Conformance to Solidity Naming Conventions
●	L09	Dead Code Elimination

STC - Succeeded Transfer Check

Criticality	minor
Location	contract.sol#L1273,1280,1297,1308

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(BMIToken).transfer(owner, fee);  
//  
IERC20(BMIToken).transfer(msg.sender, outAmount);
```

Recommendation

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

CO - Code Optimization

Criticality	minor
Location	contract.sol#L1221

Description

The contract declares “storage” variables that are used in a “view” method.

```
function pendingReward(uint256 _pid, address account) public view returns
(uint256) {
    PoolInfo storage pool = poolInfo[_pid];
    UserInfo storage user = userInfos[_pid][account];
    if (user.lastChangeTime == 0){
        return 0;
    }
    uint256 endTime = (block.timestamp>user.lockEndTime)? user.lockEndTime :
block.timestamp;
    uint256 deltaTime = (endTime > user.lastChangeTime)?
endTime.sub(user.lastChangeTime) : 0;

    uint256 rewardVal = deltaTime.mul(user.stakedAmount).mul(pool.apr).div(365
days);
    return rewardVal.div(100).add(user.rewardDebt);
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

DSRC - Data Structure Range Check

Criticality	minor
Location	contract.sol

Description

The contract could check as a precondition if the pid index is inside the pool range. This could be implemented in all the methods that accepts the pid as argument.

```
function stakeToken(uint256 _pid, uint256 _amount, address _ref) external  
nonReentrant {  
    require(!blacklist[msg.sender], "blacklist");  
    require(_amount > 0, "invalid deposit amount");  
    ...  
}
```

Recommendation

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

BA - Blacklist Addresses

Criticality	critical
Location	contract.sol#L1235,1260,1288

Description

The contract owner has the authority to blacklist addresses that have staked tokens. The owner may take advantage of it by calling the 'setBlackList' method. As a result the investors will not be able to either claim their rewards or withdraw their deposits.

```
function claim(uint256 _pid) external nonReentrant {  
    require(!blacklist[msg.sender], "blacklist");
```

Recommendation

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.

TAM - Token Address Mutation

Criticality	minor
Location	contract.sol#L1204

Description

The contract owner has the authority to change the address of the staked token. The owner may take advantage of it by calling the 'resetToken'. As a result the users will not be able to claim their rewards or withdraw their deposits since the balance of the new token address will be different from the initial.

```
function resetToken(address _token) public onlyOwner{
    BMIToken = _token;
    totalStakedAmount = 0;
}
```

Recommendation

The contract could prevent the staked token address mutation after the first initialization.

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.

FO - Fees Overflow

Criticality	critical
Location	contract.sol

Description

The properties `earlyWithdrawFee`, `earlyClaimFee` and `refFee` are used to calculate percentages values. The contract owner has the authority to set any value to these variables. If the owner sets a value higher than 100, the contract will overflow and revert the transactions.

```
function setReferralFees(uint256 _fee) public onlyOwner{
    refFee = _fee;
}

function setEarlyClaimFee(uint256 _fee) external onlyOwner {
    earlyClaimFee = _fee;
}

function setEarlyWithdrawFee(uint256 _fee) external onlyOwner {
    earlyWithdrawFee = _fee;
}
```

```
uint256 fee = outAmount.mul(earlyWithdrawFee).div(100);
outAmount = outAmount.sub(fee);
//
fee = outAmount.mul(earlyClaimFee).div(100);
outAmount = outAmount.sub(fee);
//
fee = outAmount.mul(refFee).div(100);
...
outAmount = outAmount.sub(fee);
```

Recommendation

The contract should embody a check for not allowing values greater than 100.

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.

OCTD - Owner Contract Tokens Drain

Criticality	medium
Location	contract.sol#L1349

Description

The contract owner has the authority to claim the balance that has accumulated to the staking contract from the depositors. The owner may take advantage of it by:

- Call the 'resetToken' method with an address different from the token.
- Call the 'withdrawUnknownToken' method with the token address.

```
function withdrawUnknownToken(address _token, address _address) external  
onlyOwner{  
    require(_token != BMIToken, "staking token can't be withdrawn");  
    uint256 balance = IERC20(_token).balanceOf(address(this));  
    IERC20(_token).transfer(_address, balance);  
}
```

Recommendation

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.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L91,99,793,801,818,825,832,844,867,889,912,932,1177,1181,1185,1189,1193,1204

Description

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

```
resetToken  
set  
setReferralFees  
getReceivedReferralRewards  
getPendingReferralRewards  
getReferralCounts  
decreaseAllowance  
increaseAllowance  
transferFrom  
...
```

Recommendation

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

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L71,1189,1193,1204,1209,1221,1234,1259,1287,1315,1322,1329,1333,1337,1341,1345,1349,1140

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.

```
BMIToken
_address
_token
_bool
_fee
_pid
_amount
_ref
_claimLockPeriod
...
```

Recommendation

Follow the Solidity naming convention.

<https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions>.

L09 - Dead Code Elimination

Criticality

minor

Location

contract.sol#L486,496,515,529,575,585,548,558,437,461,602,1011,988

Description

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

```
_mint  
_burn  
verifyCallResult  
sendValue  
isContract  
functionStaticCall  
functionDelegateCall  
functionCallWithValue  
functionCall  
...
```

Recommendation

Remove unused functions.

Contract Functions

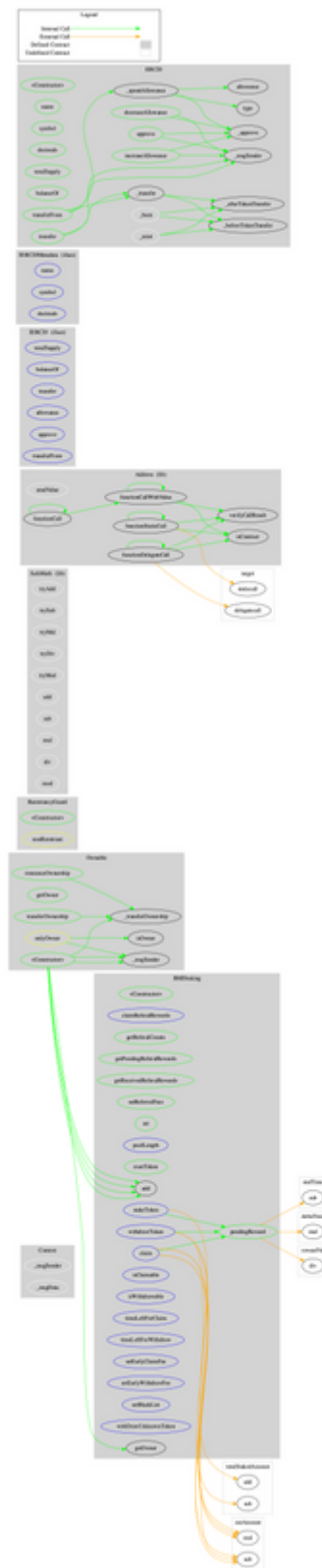
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	getOwner	Public		-
	isOwner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
ReentrancyGuard	Implementation			
	<Constructor>	Public	✓	-
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		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResult	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
ERC20	Implementation	Context, IERC20, IERC20Meta data		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-

	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
BMIStaking	Implementation	Ownable, Reentrancy Guard		
	<Constructor>	Public	✓	-
	claimReferralRewards	External	✓	-
	getReferralCounts	Public		-
	getPendingReferralRewards	Public		-
	getReceivedReferralRewards	Public		-
	setReferralFees	Public	✓	onlyOwner
	set	Public	✓	onlyOwner
	poolLength	External		-
	resetToken	Public	✓	onlyOwner
	add	Public	✓	onlyOwner
	pendingReward	Public		-
	stakeToken	External	✓	nonReentrant
	withdrawToken	External	✓	nonReentrant
	claim	External	✓	nonReentrant
	isClaimable	External		-
	isWithdrawable	External		-
	timeLeftForClaim	External		-

	timeLeftForWithdraw	External		-
	setEarlyClaimFee	External	✓	onlyOwner
	setEarlyWithdrawFee	External	✓	onlyOwner
	setBlackList	External	✓	onlyOwner
	withDrawUnknownToken	External	✓	onlyOwner

Contract Flow



Summary

Baltic Miners Staking implements a staking contract where users can deposit money and claim rewards proportional to the staking period. This audit mentions potential vulnerabilities, security concerns, business logic suggestions and potential optimizations.

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>



Cyberscope

Audit Report

Baltic Miners

May 2022

Type BEP20

Network BSC

Address 0x2df5f68db45ad4f153ce2b1447589d8c17d1306e

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Source Files	3
Audit Updates	3
Contract Analysis	4
BC - Blacklisted Contracts	5
Description	5
Recommendation	5
BT - Burn Tokens	6
Description	6
Recommendation	6
ULTW - Unlimited Liquidity to Team Wallet	7
Description	7
Recommendation	7
Contract Diagnostics	8
MTS - Manipulate Total Supply	9
Description	9
Recommendation	9
CO - Code Optimization	10
Description	10
Recommendation	10
L01 - Public Function could be Declared External	11
Description	11
Recommendation	11
L02 - State Variables could be Declared Constant	12
Description	12

Recommendation	12
L04 - Conformance to Solidity Naming Conventions	13
Description	13
Recommendation	13
L05 - Unused State Variable	14
Description	14
Recommendation	14
L09 - Dead Code Elimination	15
Description	15
Recommendation	15
L13 - Divide before Multiply Operation	16
Description	16
Recommendation	16
L14 - Uninitialized Variables in Local Scope	17
Description	17
Recommendation	17
Contract Functions	18
Contract Flow	23
Domain Info	24
Summary	25
Disclaimer	26
About Cyberscope	27

Contract Review

Contract Name	BMI
Compiler Version	v0.7.4+commit.3f05b770
Optimization	200 runs
Licence	Unlicense
Explorer	https://bscscan.com/token/0x2df5f68db45ad4f153ce2b1447589d8c17d1306e
Symbol	BMI
Decimals	5
Total Supply	3,539,520
Domain	balticminers.com

Source Files

Filename	SHA256
contract.sol	f4bbfa01900e3c3d2fd4817c0708b4feecc937f37f46935802e4b14c47c18697

Audit Updates

Initial Audit	16th May 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

BC - Blacklisted Contracts

Criticality	medium
Location	contract.sol#L589,560

Description

The contract owner has the authority to stop contracts from transactions. The owner may take advantage of it by calling the `wipeBlacklistAddress` function.

```
require(!blacklist[sender] && !blacklist[recipient], "in_blacklist");
```

The contract owner can also add specific users to a whaleis. Users that have been added in that list can not trade more tokens than the variable set as `buySellPercentForWhale`. The owner can set that variable to a very small number and prevent them from selling their tokens.

```
    if ((whalelist[sender] || whalelist[recipient]) && (pair == sender || pair == recipient)){  
  
    require(_totalSupply.mul(buySellPercentForWhale).div(10000)>amount);  
    }
```

Recommendation

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.

BT - Burn Tokens

Criticality	critical
Location	contract.sol#L872

Description

The contract owner has the authority to burn the balance of the user's tokens. The owner may take advantage of it by calling the `wipeBlacklistAddress` function.

```
function wipeBlacklistAddress(address _addr) public onlyOwner {  
    require(blacklist[_addr], "address is not blacklist");  
    uint256 _balance = _gonBalances[_addr];  
    _gonBalances[DEAD].add(_balance);  
    _gonBalances[_addr] = 0;  
    emit BlacklistAddressWiped(_addr);  
    emit Transfer(_addr, address(0x0), _balance);  
}
```

Recommendation

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#L740

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 `withdrawAllToTreasury` function.

```
function withdrawAllToTreasury(address treasury) external swapping
onlyOwner {
    uint256 amountToSwap =
    _gonBalances[address(this)].div(_gonsPerFragment);
    require( amountToSwap > 0,"There is no gen token deposited in token
contract");
    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = router.WETH();
    router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        amountToSwap,
        0,
        path,
        treasury,
        block.timestamp
    );}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile 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
●	MTS	Manipulate Total Supply
●	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
●	L09	Dead Code Elimination
●	L13	Divide before Multiply Operation
●	L14	Uninitialized Variables in Local Scope

MTS - Manipulate Total Supply

Criticality	minor
Location	contract.sol#L533

Description

Owner is able to manipulate total supply. This change will have a direct impact on the token price and Market Cap

```
for (uint256 i = 0; i < times; i++) {  
    _totalSupply = _totalSupply  
        .mul((10**RATE_DECIMALS).add(rebaseRate))  
        .div(10**RATE_DECIMALS);  
}
```

Recommendation

The contract owner should carefully manage the adjustment of the circulating supply (increases or decreases), according to the token's price fluctuations.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L1

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. The last else if statements will never be executed as the conditions are met in one of the first 2 blocks every time.

```
if (deltaTimeFromInit < (365 days)) {  
    rebaseRate = 140452;  
} else if (deltaTimeFromInit >= (365 days)) {  
    rebaseRate = 2110;  
} else if (deltaTimeFromInit >= ((15 * 365 days) / 10)) {  
    rebaseRate = 140;  
} else if (deltaTimeFromInit >= (7 * 365 days)) {  
    rebaseRate = 20;  
}
```

Recommendation

Remove the last 2 else if blocks.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L345,358,363,389,393,397,872,902,929

Description

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

```
setPairAddress  
getLiquidityBacking  
wipeBlacklistAddress  
decimals  
symbol  
name  
transferOwnership  
renounceOwnership  
owner
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality

minor

Location

contract.sol#L439,440,411,409,410,437,432,430,428,431,447,429

Description

Constant state variables should be declared constant to save gas.

```
treasuryFee  
swapEnabled  
sellFee  
liquidityFee  
genInsuranceFundFee  
firePitFee  
feeDenominator  
_symbol  
_name  
...
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L147,148,165,185,789,798,861,872,891,892,893,894,912,916,920,924,929,933,409,410,411,414,439,440,462,463,464,465,466,467

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.

```
_totalSupply  
_lastAddLiquidityTime  
_lastRebasedTime  
_initRebaseStartTime  
_autoAddLiquidity  
_autoRebase  
ZERO  
DEAD  
_isFeeExempt  
...
```

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#L13

Description

There are segments that contain unused state variables.

```
MAX_INT256
```

Recommendation

Remove unused state variables.

L09 - Dead Code Elimination

Criticality

minor

Location

contract.sol#L945,41

Description

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

```
abs  
isContract
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L514,627,902

Description

Performing divisions before multiplications may cause lose of prediction.

```
liquidityBalance = _gonBalances[pair].div(_gonsPerFragment)
_gonBalances[autoLiquidityReceiver] =
_gonBalances[autoLiquidityReceiver].add(gonAmount.div(feeDenominator).mul(_liquidityFee))
_gonBalances[address(this)] =
_gonBalances[address(this)].add(gonAmount.div(feeDenominator).mul(treasuryFee.add(genInsuranceFundFee)))
_gonBalances[firePit] =
_gonBalances[firePit].add(gonAmount.div(feeDenominator).mul(firePitFee))
feeAmount = gonAmount.div(feeDenominator).mul(_totalFee)
times = deltaTime.div(28800)
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality

minor

Location

contract.sol#L517

Description

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

```
rebaseRate
```

Recommendation

All the local scoped variables should be initialized.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	transfer	External	✓	-
	approve	External	✓	-
	transferFrom	External	✓	-
IPancakeSwap Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-

	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IPancakeSwap Router	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		-
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IPancakeSwapFactory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	Public		-
	isOwner	Public		-
	renounceOwnership	Public	✓	onlyOwner

	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
ERC20Detailed	Implementation	IERC20		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
BMI	Implementation	ERC20Detailed, Ownable		
	<Constructor>	Public	✓	ERC20Detailed
	rebase	Internal	✓	
	transfer	External	✓	validRecipient
	transferFrom	External	✓	validRecipient
	_basicTransfer	Internal	✓	
	_transferFrom	Internal	✓	
	takeFee	Internal	✓	
	addLiquidity	Internal	✓	swapping
	swapBack	Internal	✓	swapping
	withdrawAllToTreasury	External	✓	swapping onlyOwner
	shouldTakeFee	Internal		
	shouldRebase	Internal		
	shouldAddLiquidity	Internal		
	shouldSwapBack	Internal		
	setAutoRebase	External	✓	onlyOwner
	setAutoAddLiquidity	External	✓	onlyOwner
	allowance	External		-
	decreaseAllowance	External	✓	-
	increaseAllowance	External	✓	-
	approve	External	✓	-
	checkFeeExempt	External		-
	getCirculatingSupply	Public		-
	wipeBlacklistAddress	Public	✓	onlyOwner

	isNotInSwap	External		-
	manualSync	External	✓	-
	setFeeReceivers	External	✓	onlyOwner
	getLiquidityBacking	Public		-
	setWhitelist	External	✓	onlyOwner
	setBlacklist	External	✓	onlyOwner
	setWhalelist	External	✓	onlyOwner
	setBuySellPercentForWhale	External	✓	onlyOwner
	setPairAddress	Public	✓	onlyOwner
	setLP	External	✓	onlyOwner
	totalSupply	External		-
	balanceOf	External		-
	isContract	Internal		
	<Receive Ether>	External	Payable	-

Contract Flow



Domain Info

Domain Name	balticminers.com
Registry Domain ID	2676005082_DOMAIN_COM-VRSN
Creation Date	2022-02-18T13:35:33Z
Updated Date	2022-02-18T13:35:33Z
Registry Expiry Date	2023-02-18T13:35:33Z
Registrar WHOIS Server	whois.1api.net
Registrar URL	http://www.1api.net
Registrar	1API GmbH
Registrar IANA ID	1387

The domain has been created 3 months before the creation of the audit. It will expire in 9 months.

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

Summary

The Smart Contract analysis reported some issues. The contract owner has the authority to blacklist addresses and by doing that he will burn all of their token balances. The owner can also prevent specific users from transactions, and transfer the tokens accumulated by fees from the contract into the team's wallet. 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>