



Cyberscope

Audit Report

LAB ECOsystem

May 2024

Network BSC

Address 0x25331ba95a10375b765C0e2614E48621A1357D00

Audited by © cyberscope

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	MT	Mints Tokens	Unresolved
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L09	Dead Code Elimination	Unresolved
●	L15	Local Scope Variable Shadowing	Unresolved
●	L17	Usage of Solidity Assembly	Unresolved

Table of Contents

Analysis	1
Diagnostics	2
Table of Contents	3
Review	4
Audit Updates	4
Source Files	4
Findings Breakdown	5
MT - Mints Tokens	6
Description	6
Recommendation	6
L04 - Conformance to Solidity Naming Conventions	7
Description	7
Recommendation	8
L09 - Dead Code Elimination	9
Description	9
Recommendation	9
L15 - Local Scope Variable Shadowing	10
Description	10
Recommendation	10
L17 - Usage of Solidity Assembly	11
Description	11
Recommendation	11
Functions Analysis	12
Inheritance Graph	17
Flow Graph	18
Summary	19
Disclaimer	20
About Cyberscope	21

Review

Contract Name	ECOLABToken
Compiler Version	v0.6.12+commit.27d51765
Optimization	200 runs
Explorer	https://bscscan.com/address/0x25331ba95a10375b765c0e2614e48621a1357d00
Address	0x25331ba95a10375b765c0e2614e48621a1357d00
Network	BSC
Symbol	ECOLAB
Decimals	18
Total Supply	5,760,000,000
Badge Eligibility	Must Fix Criticals

Audit Updates

Initial Audit	01 May 2024
---------------	-------------

Source Files

Filename	SHA256
ECOLABToken.sol	c2ab9ac472aa4263bb8c08bd15995b91762b43eb1ccc8d5414f42fa6ee861790

Findings Breakdown



● Critical	1
● Medium	0
● Minor / Informative	4

Severity	Unresolved	Acknowledged	Resolved	Other
● Critical	1	0	0	0
● Medium	0	0	0	0
● Minor / Informative	4	0	0	0

MT - Mints Tokens

Criticality	Critical
Location	ECOLABToken.sol#L696
Status	Unresolved

Description

The `owner` address that belongs to a contract called MasterChef has the authority to mint tokens and increase the total supply up to `6000000000`. The `owner` contract may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
function mint(uint256 amount) public onlyOwner returns (bool) {  
    _mint(_msgSender(), amount);  
    return true;  
}
```

Recommendation

The team should carefully manage the functionality of the MasterChef contract. It's crucial to take additional measures to mitigate the risk associated with the mint function such as validating that the MasterChef contract cannot massively mint tokens.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	ECOLABToken.sol#L507,508,1028,1265,1270,1283
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
uint256 private constant _preMineSupply = 3720000000 * 1e18
uint256 private constant _maxSupply = 6000000000 * 1e18
address _to
uint256 _amount
address _addMinter
address _delMinter
uint256 _index
```


Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.

Find more information on the Solidity documentation

<https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention>.

L09 - Dead Code Elimination

Criticality	Minor / Informative
Location	ECOLABToken.sol#L333,338,369,398,424,434,453,467,477,759,798,990,997,1004,1018
Status	Unresolved

Description

In Solidity, dead code is code that is written in the contract, but is never executed or reached during normal contract execution. Dead code can occur for a variety of reasons, such as:

- Conditional statements that are always false.
- Functions that are never called.
- Unreachable code (e.g., code that follows a return statement).

Dead code can make a contract more difficult to understand and maintain, and can also increase the size of the contract and the cost of deploying and interacting with it.

```
function min(uint256 x, uint256 y) internal pure returns
(uint256 z) {
    z = x < y ? x : y;
}

...
```

Recommendation

To avoid creating dead code, it's important to carefully consider the logic and flow of the contract and to remove any code that is not needed or that is never executed. This can help improve the clarity and efficiency of the contract.

L15 - Local Scope Variable Shadowing

Criticality	Minor / Informative
Location	ECOLABToken.sol#L532
Status	Unresolved

Description

Local scope variable shadowing occurs when a local variable with the same name as a variable in an outer scope is declared within a function or code block. When this happens, the local variable "shadows" the outer variable, meaning that it takes precedence over the outer variable within the scope in which it is declared.

```
string memory symbol  
string memory name
```

Recommendation

It's important to be aware of shadowing when working with local variables, as it can lead to confusion and unintended consequences if not used correctly. It's generally a good idea to choose unique names for local variables to avoid shadowing outer variables and causing confusion.

L17 - Usage of Solidity Assembly

Criticality	Minor / Informative
Location	ECOLABToken.sol#L376,495,1260
Status	Unresolved

Description

Using assembly can be useful for optimizing code, but it can also be error-prone. It's important to carefully test and debug assembly code to ensure that it is correct and does not contain any errors.

Some common types of errors that can occur when using assembly in Solidity include Syntax, Type, Out-of-bounds, Stack, and Revert.

```
assembly {  
    codehash := extcodehash(account)  
}  
  
assembly {  
    let returndata_size := mload(returndata)  
    revert(add(32, returndata),  
    returndata_size)  
}  
assembly { chainId := chainid() }
```

Recommendation

It is recommended to use assembly sparingly and only when necessary, as it can be difficult to read and understand compared to Solidity code.

Functions Analysis

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
		Internal	✓	
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
		Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
IBEP20	Interface			
	totalSupply	External		-
	preMineSupply	External		-
	maxSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-

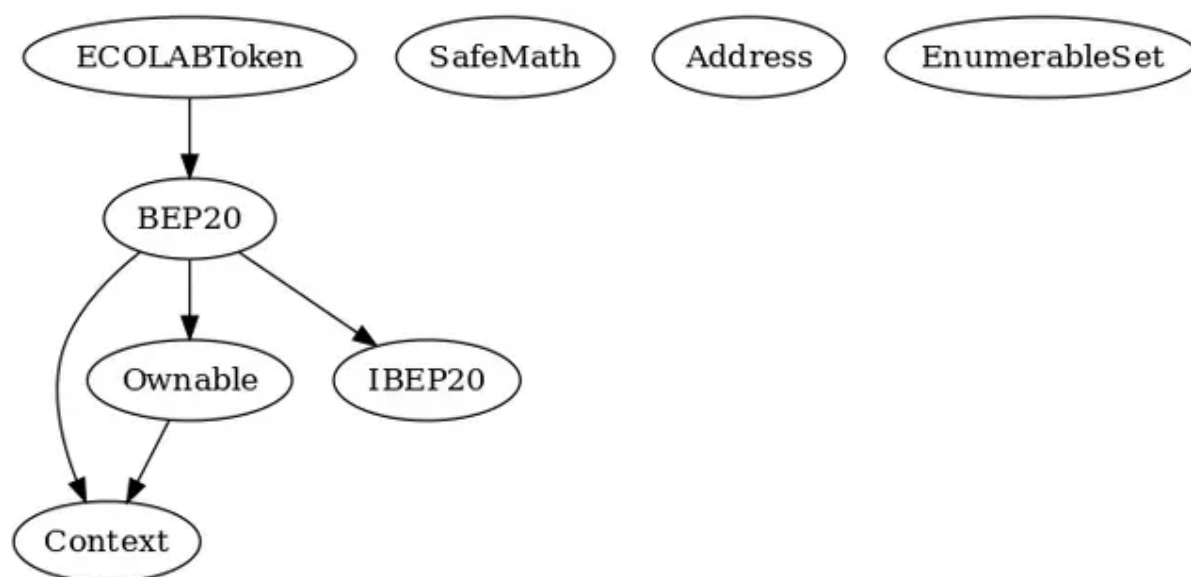
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
	min	Internal		
	sqrt	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	

	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	_functionCallWithValue	Private	✓	
BEP20	Implementation	Context, IBEP20, Ownable		
		Public	✓	-
	getOwner	External		-
	name	Public		-
	decimals	Public		-
	symbol	Public		-
	totalSupply	Public		-
	preMineSupply	Public		-
	maxSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	mint	Public	✓	onlyOwner
	_transfer	Internal	✓	

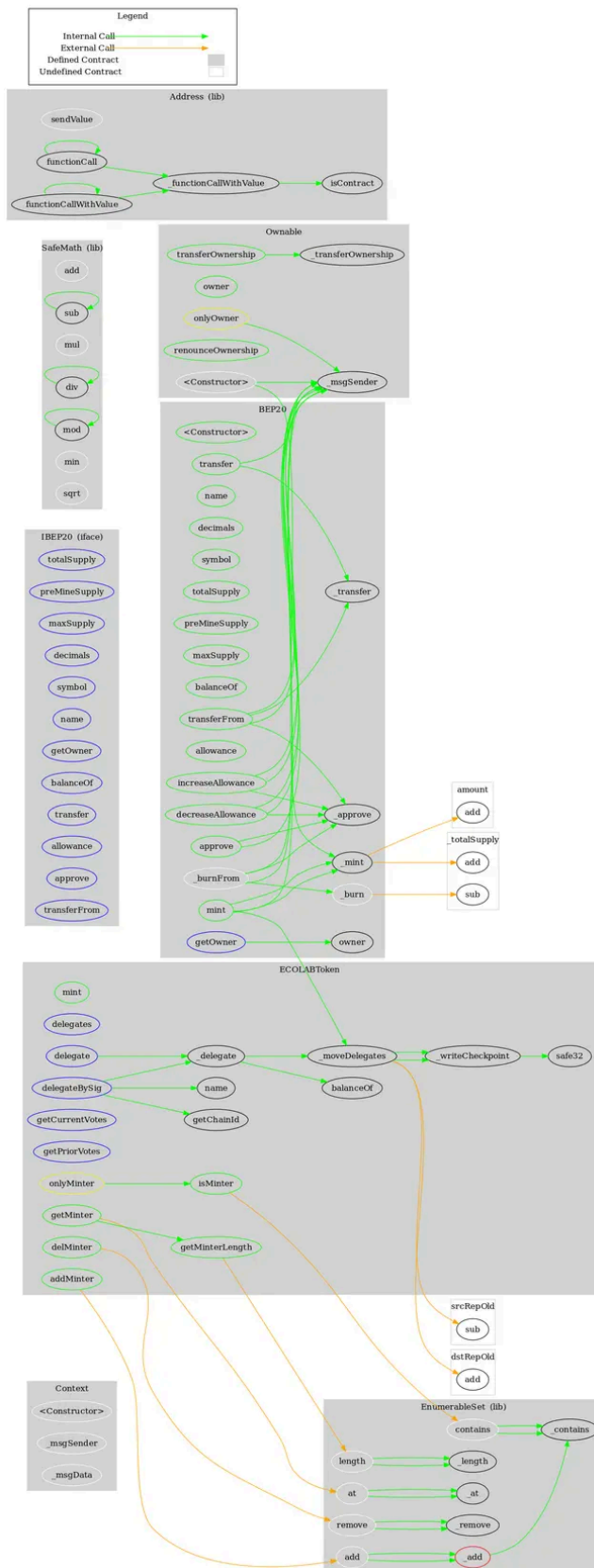
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_burnFrom	Internal	✓	
EnumerableSet	Library			
	_add	Private	✓	
	_remove	Private	✓	
	_contains	Private		
	_length	Private		
	_at	Private		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	add	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
ECOLABToken	Implementation	BEP20		

	mint	Public	✓	onlyMinter
	delegates	External		-
	delegate	External	✓	-
	delegateBySig	External	✓	-
	getCurrentVotes	External		-
	getPriorVotes	External		-
	_delegate	Internal	✓	
	_moveDelegates	Internal	✓	
	_writeCheckpoint	Internal	✓	
	safe32	Internal		
	getChainId	Internal		
	addMinter	Public	✓	onlyOwner
	delMinter	Public	✓	onlyOwner
	getMinterLength	Public		-
	isMinter	Public		-
	getMinter	Public		onlyOwner

Inheritance Graph



Flow Graph



Summary

LAB ECOsystem contract implements a token and governance mechanism. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like mint tokens. if the contract owner abuses the mint functionality, then the contract will be highly inflated. 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

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security. Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.



The Cyberscope team

<https://www.cyberscope.io>