Cyberscope

Audit Report

Karmm

November 2023

Network	BSC
Address	0x487dd60f6f9387b66922eab18a44c077fd565c94
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	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Unresolved

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	OCTD	Transfers Contract's Tokens	Unresolved
●	RVD	Redundant Variable Declaration	Unresolved
●	L19	Stable Compiler Version	Unresolved

Table of Contents

Review

Contract Name	KARMMToken
Compiler Version	v0.8.0+commit.c7dfd78e
Optimization	200 runs
Explorer	https://bscscan.com/address/0x487dd60f6f9387b66922eab18a44c077fd565c94
Address	0x487dd60f6f9387b66922eab18a44c077fd565c94
Network	BSC
Symbol	KARMM
Decimals	18
Total Supply	3,000,000,000

Audit Updates

Initial Audit	15 Nov 2023

Source Files

Filename	SHA256
KARMMToken.sol	776a945665db169e3b0949b5d5923e7b403ec973365bfce7e5a674659905c080

Findings Breakdown

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	0	0	0	0
Minor / Informative	3	0	0	0

BC - Blacklists Addresses

Criticality	Critical
Location	KARMMToken.sol#L665
Status	Unresolved

Description

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

```
   function blacklistAddress(address account) external
onlyOwner {
        require(account != address(0), "Invalid address");
        require(!isBlacklisted[account], "Address is already
blacklisted");

        isBlacklisted[account] = true;
    }
```

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.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

OCTD - Transfers Contract's Tokens

Criticality	Minor / Informative
Location	KARMMToken.sol#L656
Status	Unresolved

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `withdrawTokens` function.

```
   function withdrawTokens(address to, uint256 amount) external
onlyOwner {
        require(to != address(0), "Invalid address");
        require(amount > 0, "Amount must be greater than
zero");
        require(balanceOf(address(this)) >= amount,
"Insufficient balance in the contract");

        _transfer(address(this), to, 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.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

RVD - Redundant Variable Declaration

Criticality	Minor / Informative
Location	KARMMToken.sol#L645
Status	Unresolved

Description

The contract contains the `DECIMALS` variable that is set to `18`, which is a standard practice in ERC-20 token contracts to define the token's divisibility. However, this `DECIMALS` variable is not actively used in the contract's calculations or functions. Instead, the contract directly multiplies the `INITIAL_SUPPLY` value by `10**18`, effectively hardcoding the decimal value. This approach bypasses the need for the `DECIMALS` variable, rendering it redundant. The presence of an unused variable like `DECIMALS` not only adds unnecessary complexity to the contract but also could lead to confusion about the contract's design and its tokenomics.

```
   uint256 public constant INITIAL_SUPPLY = 3_000_000_000 *
10**18; // Total supply of 3 billion tokens
    uint8 public constant DECIMALS = 18;
```

Recommendation

It is recommended to either remove the `DECIMALS` variable from the contract if it is not being utilized in any computations or functions, or alternatively, to revise the contract to use the `DECIMALS` variable for calculating the actual `INITIAL_SUPPLY` value. This will enhance the clarity and efficiency of the contract's code. Additionally, it will prevent any potential misunderstandings or errors that might arise from the presence of an unused variable.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	KARMMToken.sol#L6,87,117,144,511,552,635
Status	Unresolved

Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.0;
```

Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

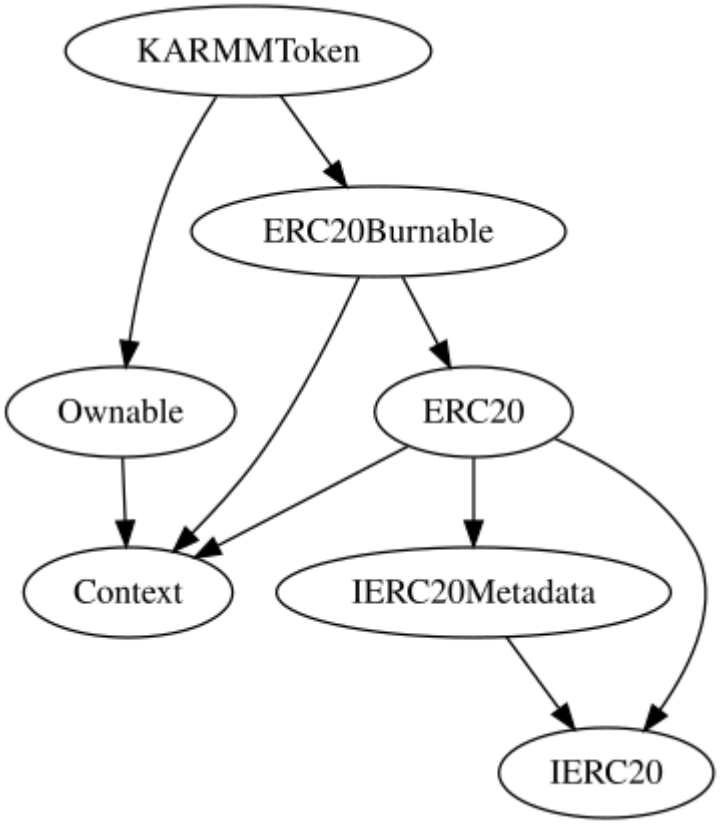
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		

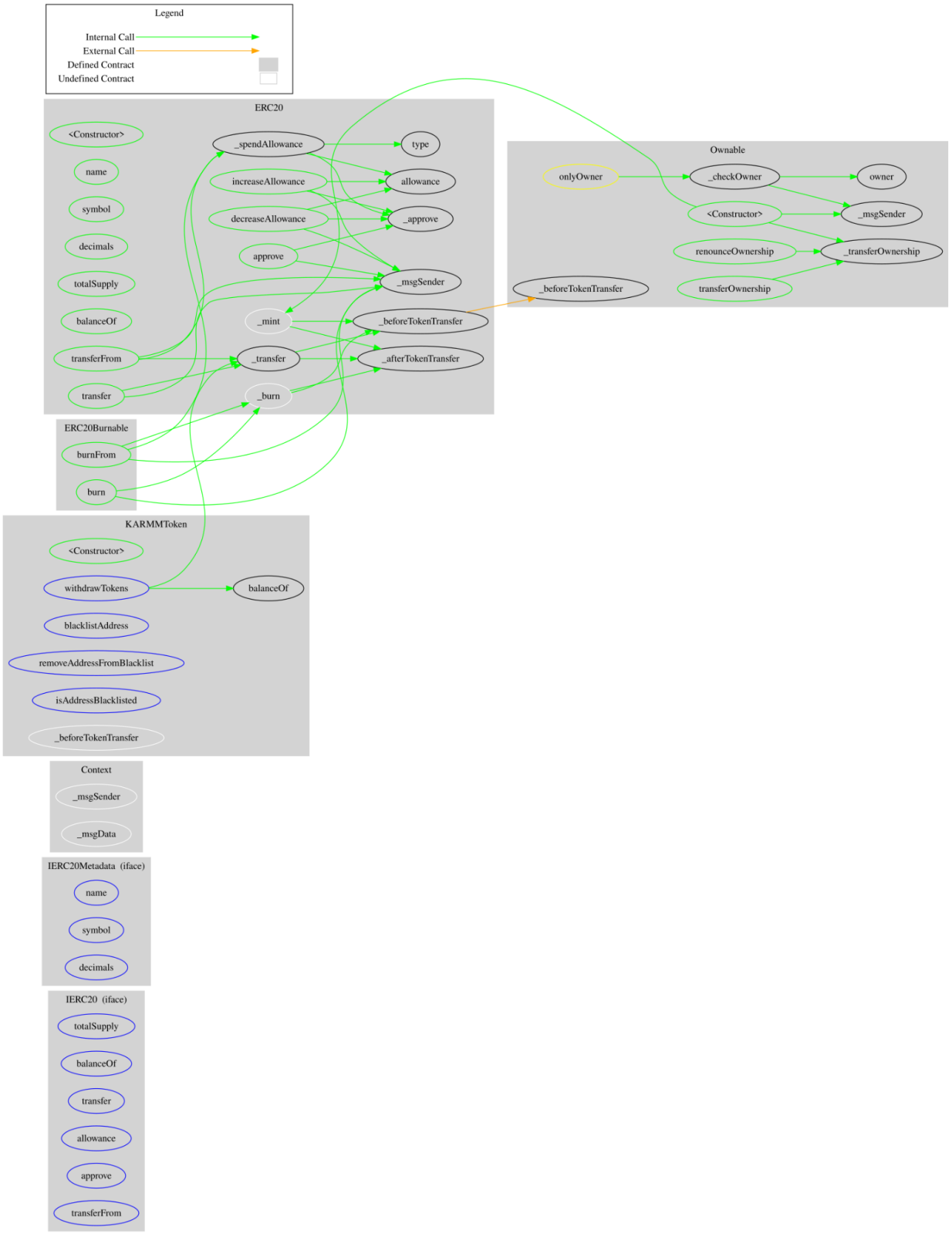
ERC20	Implementation	Context, IERC20, IERC20Metadata		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
ERC20Burnable	Implementation	Context, ERC20		

	burn	Public	✓	-
	burnFrom	Public	✓	-
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
KARMMToken	Implementation	ERC20Burnable, Ownable		
		Public	✓	ERC20
	withdrawTokens	External	✓	onlyOwner
	blacklistAddress	External	✓	onlyOwner
	removeAddressFromBlacklist	External	✓	onlyOwner
	isAddressBlacklisted	External		-
	_beforeTokenTransfer	Internal	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
transferFrom
transfer
_spendAllowance
increaseAllowance
decreaseAllowance
approve
_mint
_transfer
_burn
type
allowance
_approve
_msgSender
_beforeTokenTransfer
_afterTokenTransfer
Ownable
onlyOwner
_checkOwner
owner
<Constructor>
_msgSender
renounceOwnership
_transferOwnership
_beforeTokenTransfer
transferOwnership
ERC20Burnable
burnFrom
burn
KARMMToken
<Constructor>
withdrawTokens
balanceOf
blacklistAddress
removeAddressFromBlacklist
isAddressBlacklisted
_beforeTokenTransfer
Context
_msgSender
_msgData
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom

Summary

Karmm contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like massively blacklist addresses. 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