Cyberscope

A TAC Security Company

Audit Report

Kamirai

June 2026

Network BSC

Address 0x12bfe0251Eb64c6e26d1A264673593D6164ca213

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	Passed

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	FML	Fixed Max Limit	Acknowledged
●	IAR	Infinite Approval Reduced	Acknowledged
●	NWES	Nonconformity with ERC-20 Standard	Acknowledged
●	L19	Stable Compiler Version	Acknowledged

Table of Contents

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation**: This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation**: This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical**: Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium**: Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor**: Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative**: Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
Critical	Highly Likely / High Impact
Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
Minor / Informative	Unlikely / Low to no Impact

Review

Contract Name	Kamirai
Explorer	https://bscscan.com/address/0x12bfe0251eb64c6e26d1a264673593d6164ca213
Symbol	$KAMIRAI
Decimals	18
Total Supply	888,888,888,888

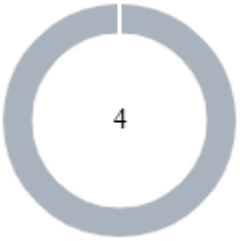
Audit Updates

Initial Audit	29 Jun 2026

Source Files

Filename	SHA256
KamiraiToken.sol	cfe79cffa7a9ab96ac4a1200d9f66c735b0e2db12319027f1d916ce874a22b5a

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	4

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

FML - Fixed Max Limit

Criticality	Minor / Informative
Location	KamiraiToken.sol#L733,802
Status	Acknowledged

Description

The max transaction limit is set for non excluded senders. Any normal address cannot transfer more than 0.5% of total supply in one transaction. Since owner is zero, this limit cannot be changed and new addresses cannot be excluded. In scenarios where an external contract, such as a launchpad factory, needs to integrate with the contract, it should be exempt from the limitations to ensure uninterrupted service and functionality. Failure to provide such exemptions can block the successful process and operation of services reliant on this contract.

```
uint256 public constant MAX_TX_AMOUNT = TOTAL_SUPPLY / 200;

require(
    amount <= MAX_TX_AMOUNT || isExcludedFromTax[sender],
    "Exceeds max tx amount"
);
```

Recommendation

It is advisable to modify the contract by incorporating functionality that enables the exclusion of designated addresses from transactional restrictions. This enhancement will allow specific addresses, such as those associated with decentralized applications (dApps) and service platforms, to operate without being hindered by the standard constraints imposed on other users.

Team Update

The team has acknowledged that this is not a security issue and states:

The max transaction limit is fixed at 0.5% of total supply per transaction for non-excluded senders. Since ownership is renounced, this limit cannot be changed and no new excluded

addresses can be added. This is a permanent and disclosed transfer-size limitation. It does not allow the team to block users or modify transfer rules; large transfers may simply need to be split into multiple transactions.

IAR - Infinite Approval Reduced

Criticality	Minor / Informative
Location	KamiraiToken.sol#L783
Status	Acknowledged

Description

The custom transferFrom reduces allowance every time, even when the allowance is set to the maximum value. This is different from the usual OpenZeppelin behavior where max approval works like infinite approval.

```
uint256 currentAllowance = allowance(sender, _msgSender());
    require(
        currentAllowance >= amount,
        "ERC20: transfer amount exceeds allowance"
    );
    _approve(sender, _msgSender(), currentAllowance - amount);
```

Recommendation

The team is advised to document that max approvals are reduced during transferFrom, unlike the usual OpenZeppelin infinite approval behavior.

Team Update

The team has acknowledged that this is not a security issue and states:

The contract reduces allowance during transferFrom even when max approval is used. This differs from the usual OpenZeppelin infinite approval behavior, but it does not give the team control over user funds and does not allow unauthorized transfers. It is an ERC-20 compatibility / user-experience note and is acknowledged by the team.

NWES - Nonconformity with ERC-20 Standard

Criticality	Minor / Informative
Location	KamiraiToken.sol#L792
Status	Acknowledged

Description

The contract does not fully conform to the ERC20 Standard. Specifically, according to the standard, transfers of 0 values must be treated as normal transfers and fire the Transfer event. However, the contract implements a conditional check that prohibits transfers of 0 values. This discrepancy between the contract's implementation and the ERC20 standard may lead to inconsistencies and incompatibilities with other contracts.

```
function _transferWithTax(
    address sender,
    address recipient,
    uint256 amount
) internal {
  ...
  require(amount > 0, "Amount must be > 0");
  ...
}
```

Recommendation

The incorrect implementation of the ERC20 standard could potentially lead to problems when interacting with the contract, as other contracts or applications that expect the ERC20 interface may not behave as expected. The team is advised to review and revise the implementation of the transfer mechanism to ensure full compliance with the ERC20 standard. https://eips.ethereum.org/EIPS/eip-20.

Team Update

The team has acknowledged that this is not a security issue and states:

The contract does not allow zero-value transfers. This is a minor ERC-20 compatibility deviation and may affect integrations that expect zero-value transfers to behave like standard ERC-20 transfers. It does not create a fund-loss vulnerability, minting risk, blacklist risk, or owner-controlled transfer risk. The team acknowledges and discloses this behavior.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	KamiraiToken.sol#L311
Status	Acknowledged

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.

```
solidity ^0.8.28;
```

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.

Team Update

The team has acknowledged that this is not a security issue and states:

The contract uses pragma solidity ^0.8.28. This is an informational best-practice note regarding compiler version locking. The deployed contract is already verified on BscScan. This item does not represent an active exploit or owner-controlled risk in the deployed contract and is acknowledged by the team.

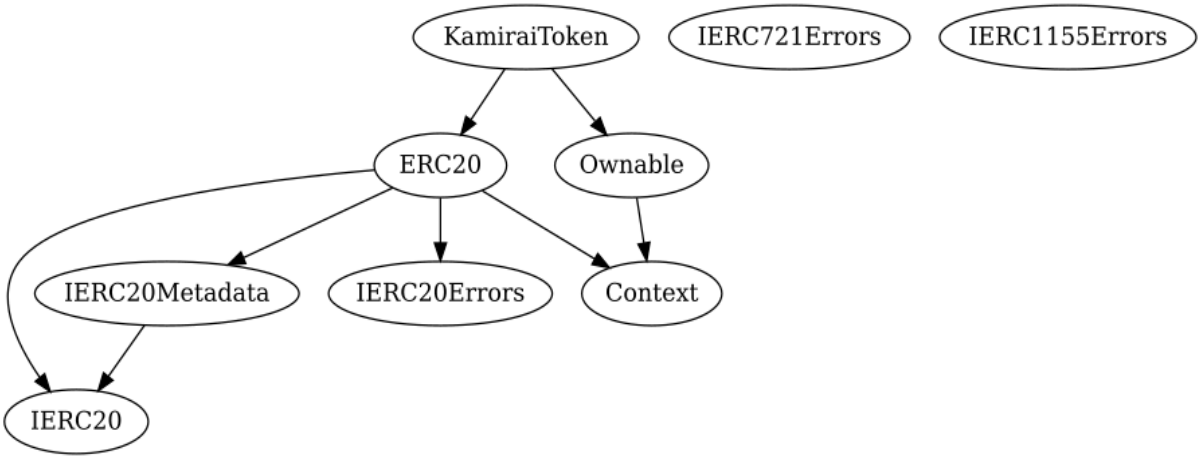
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
	_contextSuffixLength	Internal		
ERC20	Implementation	Context, IERC20, IERC20Metadata, IERC20Errors		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-

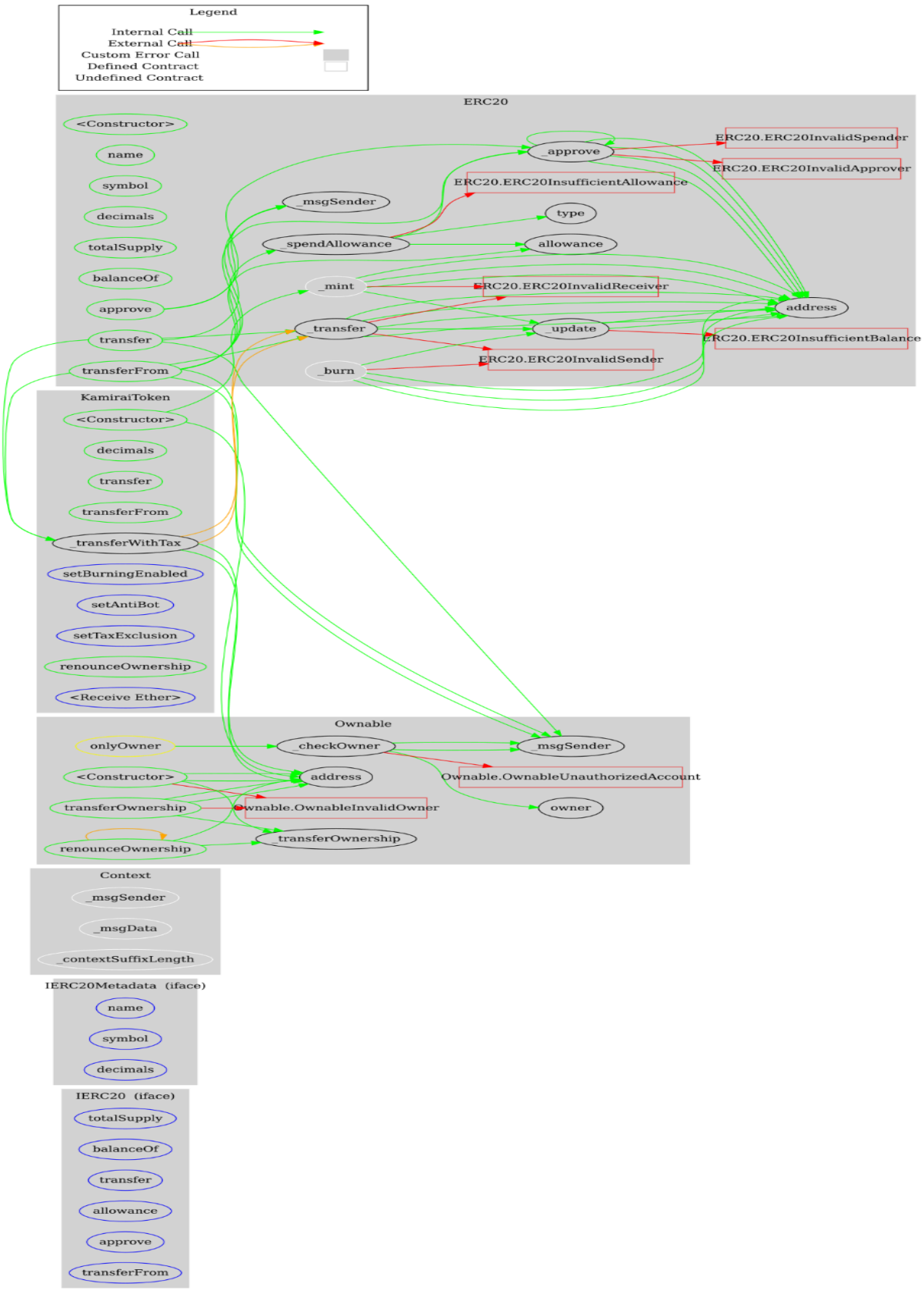
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	_transfer	Internal	✓	
	_update	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
KamiraiToken	Implementation	ERC20, Ownable		
		Public	✓	ERC20 Ownable

	decimals	Public		-
	transfer	Public	✓	-
	transferFrom	Public	✓	-
	_transferWithTax	Internal	✓	
	setBurningEnabled	External	✓	onlyOwner
	setAntiBot	External	✓	onlyOwner
	setTaxExclusion	External	✓	onlyOwner
	renounceOwnership	Public	✓	onlyOwner
		External	Payable	-

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Custom Error Call
Defined Contract
Undefined Contract
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
approve
transfer
transferFrom
_approve
ERC20.ERC20InvalidSpender
ERC20.ERC20InvalidApprover
ERC20.ERC20InsufficientAllowance
_msgSender
type
spendAllowance
allowance
_mint
ERC20.ERC20InvalidReceiver
address
_transfer
_update
ERC20.ERC20InsufficientBalance
_burn
ERC20.ERC20InvalidSender
KamiraiToken
<Constructor>
decimals
transfer
transferFrom
_transferWithTax
setBurningEnabled
setAntiBot
setTaxExclusion
renounceOwnership
<Receive Ether>
Ownable
onlyOwner
_checkOwner
_msgSender
<Constructor>
address
Ownable.OwnableUnauthorizedAccount
transferOwnership
Ownable.OwnableInvalidOwner
owner
renounceOwnership
_transferOwnership
Context
_msgSender
_msgData
_contextSuffixLength
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom

Summary

Kamirai contract implements an ERC-20 token mechanism. This audit investigated security issues, business logic concerns, and potential improvements. The smart contract analysis reported no critical or medium-severity issues. Ownership is renounced, so all owner-only administrative functions are permanently inaccessible. The team can no longer enable burn tax, enable anti-bot, update tax exclusions, or transfer ownership. Burn tax and anti-bot are currently disabled and cannot be re-enabled by the team. The remaining notes are Minor / Informative and relate to disclosed design or compatibility characteristics, including a fixed 0.5% max transaction limit, reduced infinite approval behavior, zero-value transfer handling, and compiler pragma best practice.

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 TAC 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.

A **TAC Security** Company

The Cyberscope team

cyberscope.io