Cyberscope

Audit Report

TAIGER COIN

November 2024

Network	BSC
Address	0x3E5E0fd892B0D881a7F46ECEF051c4588866c854
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
●	ISR	Inconsistent Sender Reference	Unresolved
●	MEM	Missing Error Messages	Unresolved
●	MEE	Missing Events Emission	Unresolved
●	UAR	Unexcluded Address Restrictions	Unresolved
●	UCF	Unused Code Functionality	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved

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	TaigerCoin
Compiler Version	v0.8.18+commit.87f61d96
Optimization	200 runs
Explorer	https://bscscan.com/address/0x3e5e0fd892b0d881a7f46ecef051c4588866c854
Address	0x3e5e0fd892b0d881a7f46ecef051c4588866c854
Network	BSC
Symbol	Taiger
Decimals	18
Total Supply	37,000,000,000
Badge Eligibility	Yes

Audit Updates

Initial Audit	25 Nov 2024

Source Files

Filename	SHA256
TaigerCoin.sol	3498633ef2b47d10a271980556422e696b28aec197f5766a539ae273b71d39b3

Findings Breakdown

Critical	0
Medium	0
Minor / Informative	6

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	6	0	0	0

ISR - Inconsistent Sender Reference

Criticality	Minor / Informative
Location	TaigerCoin.sol#L98
Status	Unresolved

Description

The contract is designed with a constructor that inconsistently utilizes both `msg.sender` and `_msgSender()` within the same method. This inconsistency could lead to confusion about which sender value is being referenced, especially if `_msgSender()` is overridden in a parent or inherited contract. Such discrepancies can reduce the readability and maintainability of the code and might introduce potential issues if the sender's identity is interpreted differently in various parts of the contract.

```
    constructor() {
        _balance[msg.sender] = _totalSupply;
        _isExcludedFromFees[msg.sender] = true;
        ...

        emit Transfer(address(0), _msgSender(), _totalSupply);
    }
```

Recommendation

It is recommended to streamline the code by utilizing only one approach consistently throughout the contract, either `msg.sender` or `_msgSender`. This will enhance code clarity, prevent potential misinterpretations, and ensure uniformity in sender references across the contract logic.

MEM - Missing Error Messages

Criticality	Minor / Informative
Location	TaigerCoin.sol#L59,77
Status	Unresolved

Description

The contract is missing error messages. Specifically, there are no error messages to accurately reflect the problem, making it difficult to identify and fix the issue. As a result, the users will not be able to find the root cause of the error.

```
require(isOwner())
require(newOwner != address(0))
```

Recommendation

The team is suggested to provide a descriptive message to the errors. This message can be used to provide additional context about the error that occurred or to explain why the contract execution was halted. This can be useful for debugging and for providing more information to users that interact with the contract.

MEE - Missing Events Emission

Criticality	Minor / Informative
Location	TaigerCoin.sol#L172
Status	Unresolved

Description

The contract performs actions and state mutations from external methods that do not result in the emission of events. Emitting events for significant actions is important as it allows external parties, such as wallets or dApps, to track and monitor the activity on the contract. Without these events, it may be difficult for external parties to accurately determine the current state of the contract.

```
    function ExcludeFromFees(address holder, bool exempt)
external onlyOwner {
        _isExcludedFromFees[holder] = exempt;
    }
```

Recommendation

It is recommended to include events in the code that are triggered each time a significant action is taking place within the contract. These events should include relevant details such as the user's address and the nature of the action taken. By doing so, the contract will be more transparent and easily auditable by external parties. It will also help prevent potential issues or disputes that may arise in the future.

UAR - Unexcluded Address Restrictions

Criticality	Minor / Informative
Location	TaigerCoin.sol#L186
Status	Unresolved

Description

The contract incorporates operational restrictions on transactions, which can hinder seamless interaction with decentralized applications (dApps) such as launchpads, presales, lockers, or staking platforms. 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.

```
function _transfer(
        address from,
        address to,
        uint256 amount
    ) private {
        ...
        require(amount > 1e9, "Min transfer amt");
        ...
    }
```

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. Implementing this feature will ensure smoother integration and functionality with external systems, thereby expanding the contract's versatility and effectiveness in diverse operational environments.

UCF - Unused Code Functionality

Criticality	Minor / Informative
Location	TaigerCoin.sol#L90,172
Status	Unresolved

Description

The contract includes an `_isExcludedFromFees` mapping and the associated `ExcludeFromFees` function to manage fee exemptions for specific addresses. However, this mapping and its functionality are never utilized in the contract's logic or referenced in any fee calculation processes. As a result, the mapping and the function are redundant and serve no practical purpose within the current implementation.

```
    mapping(address => bool) public _isExcludedFromFees;
      ...

     function ExcludeFromFees(address holder, bool exempt)
external onlyOwner {
          _isExcludedFromFees[holder] = exempt;
     }
```

Recommendation

It is recommended to either fully implement the `_isExcludedFromFees` mapping in the contract's logic to support its intended functionality or remove the mapping and the `ExcludeFromFees` function entirely to simplify the contract and avoid unnecessary complexity.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	TaigerCoin.sol#L86,87,88,89,94,176
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.

```
string private constant _name = "Taigercoin"
string private constant _symbol = "Taiger"
uint256 private constant _totalSupply = 37_000_000_000 * 10**18
uint8 private constant _decimals = 18
mapping(address => bool) public _isExcludedFromFees

function ExcludeFromFees(address holder, bool exempt) external
onlyOwner {
        _isExcludedFromFees[holder] = exempt;
    }
```

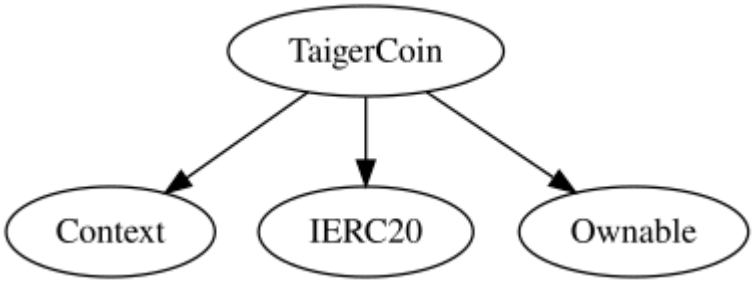
Recommendation

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/stable/style-guide.html#naming-conventions.

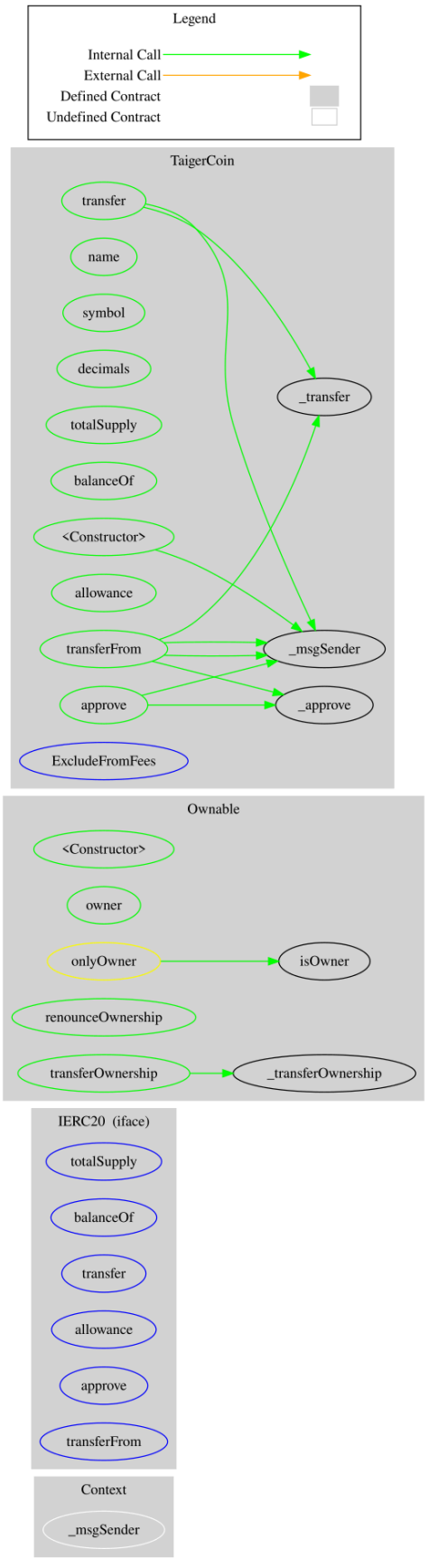
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
TaigerCoin	Implementation	Context, IERC20, Ownable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	_approve	Private	✓	
	ExcludeFromFees	External	✓	onlyOwner
	_transfer	Private	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
TaigerCoin
transfer
name
symbol
decimals
totalSupply
balanceOf
<Constructor>
allowance
transferFrom
approve
ExcludeFromFees
_transfer
_msgSender
_approve
Ownable
<Constructor>
owner
onlyOwner
renounceOwnership
transferOwnership
isOwner
_transferOwnership
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender

Summary

TAIGER COIN contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. TAIGER COIN is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

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

cyberscope.io