# Smart Contract Audit Report

Audit was conducted on the Kong City Smart Contract System

| | |
|---|---|
| Smart Contract | Code Review and Security Analysis Report for Kong City. |
| Type Of Utility | Dividend Rewards System; ERC20 Token |
| Platform | Ethereum Virtual Machine |
| Language | Solidity ^0.8.0 |
| Address | 0x9b77988b3028c3af8f863ef376ab747914f059f3 |

## Audit Score

| **Section** | **Score** |
|---|---|
| Codebase Security | **100%** |
| Codebase Complexity and Practices | **82%** |
| Owner Privileges and Control | **98%** |
| **Overall Score** | **93%** |

## Branding


Website Email Bscscan

Twitter Facebook TikTok

Telegram Instagram YouTube

# Content

# Scope of the audit

This Audit Report mainly focuses on the overall security of the Kong City Smart Contract System. This audit was conducted with rigorous attention to the general implementation of the contract and by examining the overall architectural layout of the software implementation. The reliability and correctness of this smart contract's codebase are being assessed.

## Security Scope

Identifying security related issues within each contract and the system of contract.

## General Code Quality

A full assessment of the code quality and general software architecture patterns and best practices used.

## Auditing Methods Used

Rigorous testing of the project has been performed. Detailed code base analysis was conducted, reviewing the smart contract architecture to ensure it is structured and safe.
A detailed, line by line inspection of the codebase was conducted to find any potential security vulnerabilities such as denial of service attacks, race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

Automated and manual testing was employed that included:

– Analysis of on-chain data security
– Analysis of the code in-depth and detailed, manual review of the code, line-by-line.
– Deployment of the code on an in-house testnet blockchain and running live tests.
– Determining failure preparations and if worst-case scenario protocols are in place
– Analysis of any third-party code use and verifying the overall security of this

Tools Used: Remix IDE, Ganache, Solhint, VScode, Mythril, Hardhat

# Assessing Possible Issues

Any issue detected during the conduction of this audit will be categorized under one of 3 severity levels: low, medium, and high.

## Low level Severity Issues

Issues that do not pose any serious threat to the functionality of the software

## Medium level Severity issues

Issues that can cause potential problems to the overall health of the software application but that can be fixed without having any breaking changes on the current functionality

## High level Severity issues

Critical issues that affect the smart contract's overall performance and functionality. These issues should be fixed urgently.

# General Issues Report

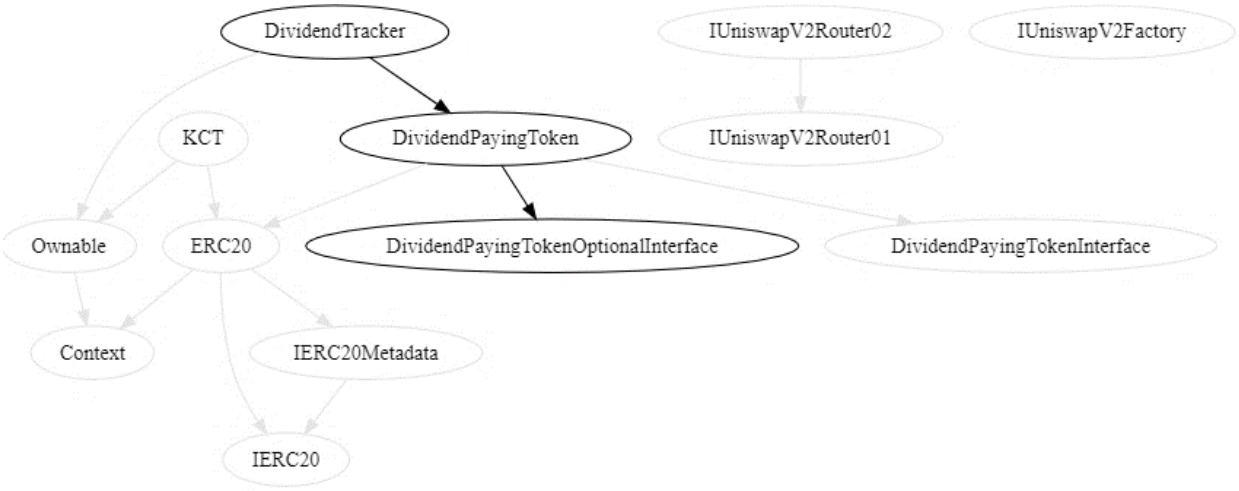
General issues that were found during manual and automatic assessments

| No | Issue Verification | Status |
|---|---|---|
| 1 | Compiler warnings | Passed |
| 2 | Reentrancy and Race Conditions. | Passed |
| 3 | Possible delays in data delivery. | Passed |
| 4 | Oracle calls. | Passed |
| 5 | DoS with block gas limit. | Passed |
| 6 | DoS with Revert. | Passed |
| 7 | Timestamp dependence. | Passed |
| 8 | Methods execution permissions. | Passed |
| 9 | Economy model. | Passed |
| 10 | Exchange impact rate on the logic. | Passed |
| 11 | Private user data leaks. | Passed |
| 12 | Scoping and Declarations. | Passed |
| 13 | Arithmetic accuracy. | Passed |

## Issues Found

| Low Level Severity | Medium Level Severity | High Level Severity |
|---|---|---|
| 0 | 0 | 0 |

# Contract Dependency Graphs




## Manual Code Inspection

The code of the target contract and its dependencies was reviewed, deployed, and manually tested by our developers.

| No | Contract | Issues |
|---|---|---|
| 1 | KCT.sol | 2 |
| 2 | DividendTracker.sol | 1 |
| 3 | DividendPayingToken.sol | 0 |

## Issues Found

| Low Level Severity | Medium Level Severity | High Level Severity |
|---|---|---|
| 0 | 3 | 0 |

# Inspections

Contract: KCT.sol
Address: TBA
Issues: 2
Notes: ERC-20 Token Implementation

### 1. Contract state can change even when logic fails

Code Line: 2364
Severity: Medium
Method: `_transfer(address from, address to,uint256 amount) internal returns (bool)`

```
2363        try
2364            dividendTracker.setBalance(payable(from), balanceOf(from))
2365        {} catch {}
2366        try dividendTracker.setBalance(payable(to), balanceOf(to)) {} catch {}
2367
```

The `setBalance` method is called within a try/catch block. This can lead to desynchronization of the dividend trackers and main contracts state.

### 2. Front Running Attack Surface

Code Line: 2470
Severity: Medium
Method: `swapTokensForEth(uint256 tokenAmount, address _to) private`

```
2469        // make the swap
2470        uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
2471            tokenAmount,
2472            0, // accept any amount of ETH
2473            path,
2474            _to,
2475            block.timestamp
2476        );
```

Setting the minimum expect output amount for a swap to be 0 can lead to frontrunning attacks that especially if there are high volume transactions involved. Calculate and set a minimum output amount or limit the max transaction amount to reduce the attack probability.
Setting the deadline parameter to the current block's timestamp may cause transactions to fail, allow a small delay when calling the router function to avoid this.

### 3. Hardcoded Gas Limit for Function Call

Code Line: 1624
Severity: Medium
Method: `_withdrawDividendOfUser(address payable user) internal returns (uint256)`

```
1624    (bool success, ) = user.call{value: _withdrawableDividend,gas: 3000}("");
1625
```

The 3000 gwei limit is relatively small and can break contract interactions with future EVM versions as operation gas costs may change with forks. We recommend calling this method using the standard syntax and avoiding using hardcoded gas limits.

## Notes

Use of the **SafeMath** library is obsolete in Solidity >= 0.8.0. The EVM performs native checks. We recommended removing the library.

The codebase is poorly organized, we recommend operating the different contracts to their own files in order to make the system more maintainable and clearer.

# Access Control and Privileges

## KCT Contract

| Role | Methods |
| --- | --- |
| Owner | updateUniswapV2Router<br>excludeFromFees<br>setAutomatedMarketMakerPair<br>updateGasForProcessing<br>updateClaimWait<br>excludeFromDividends<br>IncludeInDividends<br>setSwapAndLiquifyEnabled<br>setMarketingWallet<br>setDevWallet<br>setTeamWallet<br>setMaxWalletToken<br>updateTotalBuyTax<br>updateTotalSellTax<br>updateTaxDistributionPercentage<br>setSwapTokensAtAmount<br>setMaxBuytxAmount<br>setMaxSelltxAmount |

## Notes

The identified roles do not present any security-related risk at the time this audit was conducted.
This role may ban accounts from receiving dividends and also change the output of the taxing transactions

# DividentTracer Contract

| Role | Methods |
| --- | --- |
| Owner | renounceOwnership<br>transferOwnership<br>excludeFromDividends<br>includeInDividends<br>updateClaimWait<br>setBalance<br>processAccount |

## Notes

The identified roles do not present any security-related risk at the time this audit was conducted.4w

## Conclusion

The **Kong City** Smart contracts do not contain any high severity security issues!

## Audit Score

| Section | Score |
| --- | --- |
| Codebase Security | 100% |
| Codebase Complexity and Practices | 82% |
| Owner Privileges and Control | 98% |
| Overall Score | 93% |

# Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. To get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us based on what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Kong City and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Kong City) owe no duty of care towards you or any other person, nor does Kong City make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties, or other terms of any kind except as set out in this disclaimer, and Kong City hereby excludes all representations, warranties, conditions, and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Kong City hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Block Audit, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of the use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

KYC WITH hedgepay

SECURE . ACCESSIBLE . UNIQUE

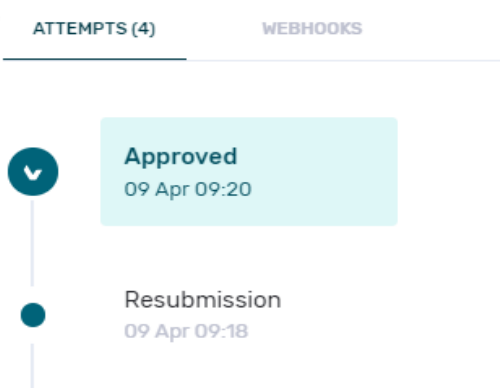
# The Owner of Kong City Has Passed All the KYC Verifications and Procedures

23:20 April 09: ID Verification 3 attempts – Passed ✓

05:28 April 10: KYC video 3 attempts – Approved ✓

06:20 April 10: Owner's contract – Verified ✓

09:12 April 10: Video Call verification – Passed ✓




*KYC PIC: Michael id:08027721*