DeHacker

Code Security Assessment

Kunci Coin

October 10th, 2022

Contents

Summary

DeHacker's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.
Possible issues we looked for included (but are not limited to):

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire code base by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes.
- Add enough unit tests to cover the possible use cases.
- Provide more comments per each function for readability, especially contracts that are verified in public.
- Provide more transparency on privileged activities once the protocol is live.

Issue Categories

Every issue in this report was assigned a severity level from the following:

Critical severity issues

A vulnerability that can disrupt the contract functioning in a number of scenarios or creates a risk that the contract may be broken.

Major severity issues

A vulnerability that affects the desired outcome when using a contract or provides the opportunity to use a contract in an unintended way.

Medium severity issues

A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.

Minor severity issues

A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.

Informational

A vulnerability that has informational character but is not affecting any of the code.

Overview

Project Summary

Project Name	Kunci Coin
Platform	bsc
website	https://kuncicoin.com/cn/
Type	Others
Deployed contract	https://bscscan.com/address/0x6cf271270662be1c4fc1b7bb7d7d7fc60cc19125#code
Language	Solidity

Audit Summary

Delivery Date	October 10th 2022
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	2	0	0	1	0	0
Medium	0	0	0	0	0	0
Minor	0	0	0	0	0	0
Informational	3	0	0	3	0	0
Discussion	0	0	0	0	0	0

Audit scope

ID	File	SHA256 Checksum
ERC	ERC20.sol	c2425e6ca50fce851f06877ccc3598651bb3b96068d4bca5d28290d3c0aea4ef

Findings

ID	Title	Category	Severity	Status
ERC-01	Centralization Related Risks	Centralization / Privilege	Major	Resolved
ERC-02	Initial Token Distribution	Centralization / Privilege	Major	Acknowledged
ERC-03	Improper usage of public and externaltype	Gas Optimization	Informational	Acknowledged
ERC-04	Too Many Digits	Coding Style	Informational	Acknowledged
ERC-05	Unlocked Compiler Version	Language Specific	Informational	Acknowledged

Major

ERC-01 | Centralization Related Risks

Category	Severity	Location	Status
Centralization / Privilege	Major	ERC20.sol	Resolved

Description

In the contract Ownable , the role owner has authority over the following functions:

function renounceOwnership()
function transferOwnership(address newOwner)

In the contract ERC20 , the role owner has authority to mint tokens to anyone and burn anyone's tokensover the following functions:

function mint(address account, uint256 amount)
function burn(address account, uint256 amount)

Any compromise to the owner account may allow a hacker to take advantage of this authority.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the securityoperation and level of decentralization, which in most cases cannot be resolved entirely at the presentstage. We advise the client to carefully manage the privileged account's private key to avoid any potentialrisks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol beimproved via a decentralized mechanism or smart-contract-based accounts with enhanced securitypractices, e.g., multi-signature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a differentlevel in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (2/3,3/5) combination mitigate by delaying the sensitive operation and avoiding asingle point of key management failure.

Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
AND
Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to theprivate key compromised;
AND
A medium/blog link for sharing the timelock contract and multi-signers addresses information withthe public audience.

Long Term:
Timelock and DAO, the combination, mitigate by applying decentralization and transparency.

Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
AND
Introduction of a DAO/governance/voting module to increase transparency and user involvement;
AND
A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO informationwith the public audience.

Permanent:
Renouncing the ownership or removing the function can be considered fully resolved.
Renounce the ownership and never claim back the privileged roles;
OR
Remove the risky functionality.

Noted: Recommend considering the long-term solution or the permanent solution. The project team shallmake a decision based on the current state of their project, timeline, and project resources.

Alleviation

The development team has renounced the ownership through transaction 0xaa6bf71bf4a9a62ce9643e2bebec3e4a58e0b0d6120e671a3be9bd3575baf6c9.

Informational

ERC-02 | Initial Token Distribution

Category	Severity	Location	Status
Centralization / Privilege	Major	ERC20.sol: 242	Acknowledged

Description

All of the tokens are sent to the contract deployer when deploying the contract. This could be acentralization risk as the deployer can distribute the tokens without obtaining the consensus of thecommunity.

Recommendation

We recommend the team to be transparent regarding the initial token distribution process, and the teamshall make enough efforts to restrict the access of the private key.

Alleviation

The development team responded that initial token distribution is laid out in their whitepaper and they areemploying a multi-sig strategy to restrict the access of the private key.

Informational

ERC-03 | Improper Usage Of public And external Type

Category	Severity	Location	Status
Gas Optimization	Informational	ERC20.sol: 174, 182, 250, 258, 282, 289, 301, 309, 320 , 338, 366, 385, 440, 463	Acknowledged

Description

public functions that are never called by the contract could be declared as external . externalfunctions are more efficient than public functions.

Recommendation

Consider using the external attribute for public functions that are never called within the contract.

Alleviation

The team acknowledged the finding and they will leave it as it is for now.

Informational

ERC-04 | Too Many Digits

Category	Severity	Location	Status
Coding Style	Informational	ERC20.sol: 241	Acknowledged

Description

Literals with many digits are difficult to read and review.

```
        _totalSupply = 100000000000 * (10 ** uint256(decimals()));
```

Recommendation

We advise the client to use the scientific notation to improve readability.

Alleviation

The team acknowledged the finding and they will leave it as it is for now.

Informational

ERC-05 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	ERC20.sol: 6	Acknowledged

Description

The contract has unlocked compiler version. An unlocked compiler version in the source code of thecontract permits the user to compile it at or above a particular version. This, in turn, leads to differences inthe generated bytecode between compilations due to different compiler versions. This can lead to anambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard toidentify over a span of multiple compiler versions rather than a specific one.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the contract canbe compiled at. For example, for version v0.8.0 the contract should contain the following line:

```
pragma solidity 0.8.0;
```

Alleviation

The team acknowledged the finding and they will leave it as it is for now.

Disclaimer

This report is subject to the terms and conditions (including but not limited to the description of the Services, confidentiality, disclaimers and limitations of liability) or scope of the Services set forth in the Service Agreement and the terms and conditions provided to you (the "Customer" or the "Company") in connection with this Agreement. This report relating to the Services set forth herein shall be used by the Company only to the extent permitted by the terms and conditions set forth herein. No one may transmit, disclose, quote or rely on this report for any purpose, nor may a copy be delivered to any person other than the Company, without Dehacker' prior written consent.

This report is not and should not be construed as an "endorsement" or "disapproval" of any particular project or team. This report is not and should not be construed as an indication of the economics or value of any product or asset created by any team or program contracted by Dehacker for the safety assessment. This report does not provide any warranties or warranties as to the absolutely defect-free nature of the analyzed technology, nor does it provide any indication of the technology owner, business, business model, or legal compliance.

This report should not be used in any way to make decisions surrounding investment or participation in any particular project. This report in no way provides investment advice and should not be used as investment advice of any kind. This report represents a broad evaluation process designed to help our customers improve the quality of their code while reducing the high risks posed by cryptographic tokens and blockchain technology.

Blockchain technology and crypto assets have a high level of ongoing risk. Dehacker' position is that each company and individual is responsible for their own due diligence and ongoing safety. The goal of Dehacker is to help reduce the medium of attack and the high level of variance associated with utilizing new and changing technologies, and in no way guarantee the safety or functionality of the technologies we agree to analyze.

The assessment service provided by Dehacker is influenced by dependencies and is under continued development. You agree that your access and/or use, including but not limited to any services, reports and materials, will be at your own risk as is, as is and as available. Cryptographic tokens are an emerging technology and carry a high level of technical risk and uncertainty. Evaluation reports may include false positives, false negatives, and other unpredictable results. These services can access and rely on multiple layers of third parties.

ALL SERVICES, LABELS, EVALUATION REPORTS, WORK PRODUCTS OR OTHER MATERIALS, OR ANY PRODUCT OR USE RESULTS ARE PROVIDED "AS IS" AND "AVAILABLE" WITHOUT WARRANTIES OF ANY KIND FOR FAULTS AND DEFECTS. TO THE FULLEST EXTENT PERMITTED BY APPLICABLE LAW, Dehacker DISCLAIMS ANY

WARRANTY, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, FOR THE SERVICES, EVALUATION REPORTS OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, DehackerS EXPRESSLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM THE TRANSACTION, USE OR TRADE PRACTICES. WITHOUT LIMITING THE FOREGOING, Dehacker DON'T GUARANTEE SERVICE, TAG, ASSESSMENT, WORK PRODUCTS, OR OTHER MATERIAL, OR THE USE OF ANY PRODUCT OR RESULT WILL MEET THE REQUIREMENTS OF CUSTOMERS OR ANY OTHERS, REALIZATION OF THE EXPECTED RESULTS OF ANY COMPATIBLE WITH ANY SOFTWARE, SYSTEM OR OTHER SERVICES, OR SAFE, ACCURATE AND COMPLETE, NO HARMFUL CODE OR NO ERROR. WITHOUT LIMITING THE FOREGOING, Dehacker DOESN'T PROVIDE ANY GUARANTEE OR PROMISE, ALSO DO NOT MAKE ANY STATEMENT, AS A SERVICE TO MEET THE REQUIREMENTS OF CUSTOMERS, IMPLEMENT ANY EXPECTED RESULTS, AND ANY OTHER SOFTWARE, APPLICATION, SYSTEM OR SERVICE COMPATIBLE OR WORK TOGETHER, CONTINUOUS OPERATION, TO MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR ERROR-FREE, OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER Dehacker NOR ANY Dehacker AGENT MAKES ANY REPRESENTATION OR WARRANTY, EXPRESS OR IMPLIED, AS TO THE ACCURACY, RELIABILITY OR TIMELINESS OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICES. Dehacker SHALL NOT BE LIABLE OR LIABLE FOR ANY ERRORS, ERRORS OR INACCURACIES IN THE CONTENT AND MATERIALS, OR FOR ANY LOSS OR DAMAGE OF ANY KIND ARISING OUT OF THE USE OF ANY CONTENT, OR (II) FOR PERSONAL INJURY OR PROPERTY DAMAGE OF ANY NATURE ARISING OUT OF CUSTOMER'S ACCESS TO OR USE OF THE SERVICES. EVALUATION REPORT OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATIONS OR WARRANTIES OF ANY THIRD-PARTY MATERIALS OR RELATING TO ANY THIRD-PARTY MATERIALS ARE STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNERS OR DISTRIBUTORS OF THE THIRD-PARTY MATERIALS.

THE SERVICES, EVALUATION REPORTS AND ANY OTHER MATERIALS UNDER THIS AGREEMENT ARE PROVIDED TO THE CUSTOMER ONLY AND SHALL NOT BE RELIED UPON BY ANY COMPANY

COPIES MAY NOT BE DELIVERED TO ANY OTHER PERSON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT EXPRESSLY SPECIFIED IN THIS AGREEMENT WITHOUT Dehacker 'PRIOR WRITTEN CONSENT.

NO THIRD PARTY OR ANY PERSON ACTING ON BEHALF OF ANY THIRD PARTY SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, EVALUATION REPORTS AND ANY ACCOMPANYING MATERIALS, AND NO SUCH THIRD PARTY SHALL BE ENTITLED TO MAKE ANY CONTRIBUTION TO Dehacker WITH RESPECT TO SUCH SERVICES, EVALUATION REPORTS AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF Dehacker CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF THE CUSTOMER. THEREFORE, NO THIRD PARTY OR ANY PERSON

ACTING ON BEHALF OF ANY SUCH REPRESENTATIONS AND WARRANTIES SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES, AND NO SUCH THIRD PARTY SHALL BE ENTITLED TO MAKE ANY CONTRIBUTION TO Dehacker WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER THAT IS BOUND BY THIS AGREEMENT OR OTHERWISE OR WHICH GIVES RISE TO INDEMNIFICATION.

FOR THE AVOIDANCE OF DOUBT, THE SERVICES (INCLUDING ANY RELATED EVALUATION REPORTS OR MATERIALS) SHOULD NOT BE REGARDED OR RELIED UPON FOR FINANCIAL, TAX, LEGAL, REGULATORY OR OTHER ADVICE OF ANY KIND.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Coding Style

Coding Style findings usually do not affect the generated bytecode but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block. timestamp works.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux “sha256sum" command against the target file.

About

DeHacker is a team of auditors and white hat hackers who perform security audits and assessments. With decades of experience in security and distributed systems, our experts focus on the ins and outs of system security. Our services follow clear and prudent industry standards. Whether it's reviewing the smallest modifications or a new platform, we'll provide an in-depth security survey at every stage of your company's project. We provide comprehensive vulnerability reports and identify structural inefficiencies in smart contract code, combining high-end security research with a real-world attacker mindset to reduce risk and harden code.

BLOCKCHAIINS

Ethereum

Cosmos

Eos

Substrate

TECH STACK

Python

Solidity

Rust

c++

CONTACTS

https://dehacker.io

https://twitter.com/dehackerio

https://github.com/dehacker/audits_public

https://t.me/dehackerio

https://blog.dehacker.io/

audit@dehacker.io

DeHacker

October 2022