CERTIK

Security Assessment

Cojam Token

May 29th, 2021

Summary

This report has been prepared for Cojam Token smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Dynamic Analysis, Static Analysis, and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- 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 codebase 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 given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

No notable vulnerabilities were identified in the codebase and it makes use of the latest security principles and style guidelines. There were certain optimizations observed as well as security principles that can optionally be applied to the codebase to fortify the codebase to a greater extent.

Overview

Project Summary

Project Name	Cojam Token
Description	A typical ERC-20 token with enhanced features.
Platform	Klaytn
Language	Solidity
Codebase	https://github.com/cojam-limited/audit-Cojam
Commits	1. 534159f5059b48b956b511e832ea26776a207416 2. 9dc7d06f9db0655c0ee4cb7af5150e98bee9c62e

Audit Summary

Delivery Date	May 29, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	ERC-20 Token

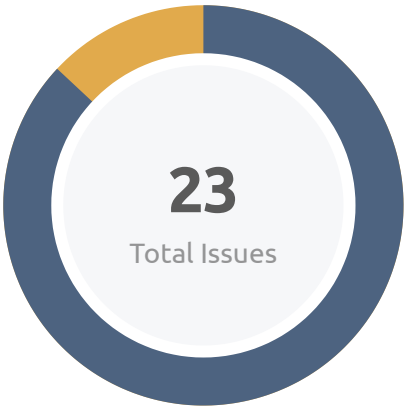
Vulnerability Summary

Total Issues	23
Critical	0
Major	0
Minor	3
Informational	20
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
CTC	contracts/CojamToken.sol	eb6414a5d85dc73c9808bc1b9948e871c684e87ce4d19df845c76e64e11d3e49
KIP	contracts/kip7/KIP7.sol	5bf7fbcb7c7ce68eebcbd021a9f360cb047fe7c1aac84286851bf0764d4e7e5c
KIB	contracts/kip7/KIP7Burnable.sol	5cc412eb02fecfbf912684d9c8e090425b030f1279075ced5ec0056ab786bbdd
KIL	contracts/kip7/KIP7Lockable.sol	c80d491441ddd0862b79cfeace5111acebfc1f6d9e9d3749290c833e4e81ec29
FCD	contracts/library/Freezable.sol	dbe169a09f2808a7d14bfab1e7098829dc82f1d1e3b37f82ff0a6a937346189c
OCD	contracts/library/Ownable.sol	283f3f4b91e0c11a5563c816653f476b7e19849b745dd315f80b7636eedf17a7
PCD	contracts/library/Pausable.sol	8833ef45617680101a26560939c74262a21a5625f4463944970567548b628b5b
SMC	contracts/library/SafeMath.sol	87f0cd89268383b3db9b1cfb26a3a45494492dc77bbf5f47ff45b69eeb63be6e

Findings



Critical	0 (0.00%)
Major	0 (0.00%)
Minor	3 (13.04%)
Informational	20 (86.96%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
CTC-01	User-Defined Getters	Gas Optimization	Informational	Acknowledged
CTC-02	Ignored `return` Value	Inconsistency	Informational	Partially Resolved
CTC-03	Race Condition	Volatile Code	Minor	Acknowledged
CTC-04	Omitted SPDX License Identifier	Language Specific	Informational	Resolved
FCD-01	Return Variable Utilization	Gas Optimization	Informational	Resolved
FCD-02	Omitted SPDX License Identifier	Language Specific	Informational	Resolved
KIB-01	Ignored `return` Value	Inconsistency	Informational	Partially Resolved
KIB-02	Ambiguous Statement	Volatile Code	Informational	Resolved
KIB-03	Ambiguous `return` Value	Volatile Code	Informational	Resolved
KIB-04	Omitted SPDX License Identifier	Language Specific	Informational	Resolved
KIL-01	Ignored `return` Value	Inconsistency	Informational	Partially Resolved
KIL-02	Return Variable Utilization	Gas Optimization	Informational	Resolved
KIL-03	Pattern Inconsistency	Inconsistency	Informational	Acknowledged
KIL-04	Omitted SPDX License Identifier	Language Specific	Informational	Resolved
KIP-01	User-Defined Getters	Gas Optimization	Informational	Acknowledged
KIP-02	Omitted SPDX License Identifier	Language Specific	Informational	Resolved

ID	Title	Category	Severity	Status
OCD-01	User-Defined Getters	Gas Optimization	Informational	Acknowledged
OCD-02	Pull-Over-Push Pattern	Logical Issue	Minor	Acknowledged
OCD-03	Potential `admin`-less Contract	Logical Issue	Minor	Acknowledged
OCD-04	Omitted SPDX License Identifier	Language Specific	Informational	Resolved
PCD-01	User-Defined Getters	Gas Optimization	Informational	Acknowledged
PCD-02	Omitted SPDX License Identifier	Language Specific	Informational	Resolved
SMC-01	Omitted SPDX License Identifier	Language Specific	Informational	Resolved

CTC-01 | User-Defined Getters

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/CojamToken.sol: 14~16	Acknowledged

Description

The linked variables contain user-defined getter functions that are equivalent to their name barring for an underscore (`_`) prefix / suffix.

Recommendation

We advise that the linked variables are instead declared as `public` and that they are renamed to their respective getter's name as compiler-generated getter functions are less prone to error and much more maintainable than manually written ones.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

CTC-02 | Ignored `return` Value

Category	Severity	Location	Status
Inconsistency	Informational	contracts/CojamToken.sol: 20, 35, 51, 52~59, 72	Partially Resolved

Description

The linked function invocations do not check the return value of the function call which should yield a `true` result.

Recommendation

We advise to remove the `return` values from the internal functions.

Alleviation

The development team opted to consider our references and adjusted the majority of the linked functions to utilize the `return` values.

CTC-03 | Race Condition

Category	Severity	Location	Status
Volatile Code	Minor	contracts/CojamToken.sol: 63~74	Acknowledged

Description

The token suffers from a known race conditional where:

- Alice allows Bob to transfer N of Alice's tokens (N>0) by calling approve method on Token smart contract passing Bob's address and N as method arguments.
- After some time, Alice decides to change from N to M (M>0) the number of Alice's tokens Bob is allowed to transfer, so she calls approve method again, this time passing Bob's address and M as method arguments.
- Bob notices Alice's second transaction before it was mined and quickly sends another transaction that calls transferFrom method to transfer N Alice's tokens somewhere.
- If Bob's transaction will be executed before Alice's transaction, then Bob will successfully transfer N Alice's tokens and will gain an ability to transfer another M tokens.
- Before Alice noticed that something went wrong, Bob calls transferFrom method again, this time to transfer M Alice's tokens.

So, Alice's attempt to change Bob's allowance from N to M (N>0 and M>0) made it possible for Bob to transfer N+M of Alice's tokens, while Alice never wanted to allow so many of her tokens to be transferred by Bob.

Recommendation

We advise to implement `increaseAllowance()` and `decreaseAllowance()` functions to update the allowance of a user and only use `approve()` when the allowance is zero.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

CTC-04 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/CojamToken.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

FCD-01 | Return Variable Utilization

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/library/Freezable.sol: 35	Resolved

Description

The linked function declarations contain explicitly named `return` variables that are not utilized within the function's code block.

Recommendation

We advise that the linked variables are either utilized or omitted from the declaration.

Alleviation

The development team opted to consider our references and removed the redundant return variables.

FCD-02 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/library/Freezable.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

KIB-01 | Ignored `return` Value

Category	Severity	Location	Status
Inconsistency	Informational	contracts/kip7/KIP7Burnable.sol: 26	Partially Resolved

Description

The linked function invocations do not check the return value of the function call which should yield a `true` result.

Recommendation

We advise to remove the `return` values from the internal functions.

Alleviation

The development team opted to consider our references and adjusted the majority of the linked functions to utilize the `return` values.

KIB-02 | Ambiguous Statement

Category	Severity	Location	Status
Volatile Code	Informational	contracts/kip7/KIP7Burnable.sol: 18	Resolved

Description

The linked statement updates the `success` local variable despite its updated value in L16.

Recommendation

We advise to remove the `return` values from the `internal` functions.

Alleviation

The development team opted to consider our references and used the `success` local variable in the `return` statement.

KIB-03 | Ambiguous `return` Value

Category	Severity	Location	Status
Volatile Code	Informational	contracts/kip7/KIP7Burnable.sol: 28~35	Resolved

Description

The returned value of the `burnFrom()` function is not composed by the result of its two `internal` mechanisms.

Recommendation

We advise to directly update the `success` local variable to `true` at the linked function.

Alleviation

The development team opted to consider our references and adjusted the `burnFrom()` function to require both of the `internal` functionalities it utilizes to return `true`.

KIB-04 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/kip7/KIP7Burnable.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

KIL-01 | Ignored `return` Value

Category	Severity	Location	Status
Inconsistency	Informational	contracts/kip7/KIP7Lockable.sol: 51, 87, 88	Partially Resolved

Description

The linked function invocations do not check the return value of the function call which should yield a `true` result.

Recommendation

We advise to remove the `return` values from the internal functions.

Alleviation

The development team opted to consider our references and adjusted the majority of the linked functions to utilize the `return` values.

KIL-02 | Return Variable Utilization

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/kip7/KIP7Lockable.sol: 49	Resolved

Description

The linked function declarations contain explicitly named `return` variables that are not utilized within the function's code block.

Recommendation

We advise that the linked variables are either utilized or omitted from the declaration.

Alleviation

The development team opted to consider our references and removed the redundant return variables.

KIL-03 | Pattern Inconsistency

Category	Severity	Location	Status
Inconsistency	Informational	contracts/kip7/KIP7Lockable.sol: 29	Acknowledged

Description

The `due` date is checked in the internal `lock` functionality, but the pattern is then reversed, as the `unlock` function, which checks the `due` date in its `external` function instead of its `internal` one.

Recommendation

We advise to keep a consistent codebase, hence increasing its legibility.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

KIL-04 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/kip7/KIP7Lockable.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

KIP-01 | User-Defined Getters

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/kip7/KIP7.sol: 8	Acknowledged

Description

The linked variables contain user-defined getter functions that are equivalent to their name barring for an underscore (`_`) prefix / suffix.

Recommendation

We advise that the linked variables are instead declared as `public` and that they are renamed to their respective getter's name as compiler-generated getter functions are less prone to error and much more maintainable than manually written ones.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

KIP-02 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/kip7/KIP7.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

OCD-01 | User-Defined Getters

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/library/Ownable.sol: 4	Acknowledged

Description

The linked variables contain user-defined getter functions that are equivalent to their name barring for an underscore (`_`) prefix / suffix.

Recommendation

We advise that the linked variables are instead declared as `public` and that they are renamed to their respective getter's name as compiler-generated getter functions are less prone to error and much more maintainable than manually written ones.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

OCD-02 | Pull-Over-Push Pattern

Category	Severity	Location	Status
Logical Issue	Minor	contracts/library/Ownable.sol: 28~35	Acknowledged

Description

The change of admin overrides the previously set admin with the new one without guaranteeing the new admin is able to actuate transactions on-chain.

Recommendation

We advise the pull-over-push pattern to be applied here whereby a new owner is first proposed and consequently needs to accept the owner status ensuring that the account can actuate transactions on-chain.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

OCD-03 | Potential `admin`-less Contract

Category	Severity	Location	Status
Logical Issue	Minor	contracts/library/Ownable.sol: 37~39	Acknowledged

Description

The `renounceOwnership()` function allows for an `admin`-less contract.

Recommendation

We advise to adjust the linked function by checking that a temporary admin is in place.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

OCD-04 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/library/Ownable.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

PCD-01 | User-Defined Getters

Category	Severity	Location	Status
Gas Optimization	Informational	contracts/library/Pausable.sol: 6	Acknowledged

Description

The linked variables contain user-defined getter functions that are equivalent to their name barring for an underscore (`_`) prefix / suffix.

Recommendation

We advise that the linked variables are instead declared as `public` and that they are renamed to their respective getter's name as compiler-generated getter functions are less prone to error and much more maintainable than manually written ones.

Alleviation

The development team has acknowledged this exhibit but decided to not apply its remediation in the current version of the codebase.

PCD-02 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/library/Pausable.sol: 1	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

SMC-01 | Omitted SPDX License Identifier

Category	Severity	Location	Status
Language Specific	Informational	contracts/library/SafeMath.sol: 2	Resolved

Description

The SPDX license identifier not provided in source file.

Recommendation

Before publishing, consider adding one such comment to each source file.

Alleviation

The development team opted to consider our references and added SPDX license identifiers to all the smart contracts in the codebase.

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

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.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

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

Data Flow

Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a struct assignment operation affecting an in-memory struct rather than an in-storage one.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Coding Style

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

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers

Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error

Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK'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 considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report 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 in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK'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.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.