PUBLIC

Code Assessment

of the MCAG Contracts

Smart Contracts

October 4, 2023

Produced for

by

Contents

1 Executive Summary

Dear all,

Thank you for trusting us to help Mimo Capital AG with this security audit. Our executive summary provides an overview of subjects covered in our audit of the latest reviewed contracts of MCAG Contracts according to Scope to support you in forming an opinion on their security risks.

Mimo Capital AG issues ERC-721 compliant NFTs called KUMABondTokens, which are backed by real-world bonds. Additional smart contracts handle functionalities like KYC compliance through KYCToken, role-based access control via AccessController, and price feed updates through MCAGAggregator and KIBTAggregator. The system also allows for pausing the tokens and maintains a blacklist of addresses that cannot interact with the KUMABondTokens.

The most critical subjects covered in our audit are functional correctness and access control. Security regarding all the aforementioned subjects is high.

The general subjects covered are code complexity, suitability of the implementation for the intended use case and accuracy of the documentation.

In summary, we find that the codebase provides a good level of security.

It is important to note that security audits are time-boxed and cannot uncover all vulnerabilities. They complement but don't replace other vital measures to secure a project.

The following sections will give an overview of the system, our methodology, the issues uncovered and how they have been addressed. We are happy to receive questions and feedback to improve our service.

Sincerely yours,

ChainSecurity

1.1 Overview of the Findings

Below we provide a brief numerical overview of the findings and how they have been addressed.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	1
• Code Corrected	1

2 Assessment Overview

In this section, we briefly describe the overall structure and scope of the engagement, including the code commit which is referenced throughout this report.

2.1 Scope

The assessment was performed on the source code files inside the MCAG Contracts repository based on the documentation files. The table below indicates the code versions relevant to this report and when they were received.

V	Date	Commit Hash	Note
1	24 August 2023	d3e796a8b0e40d5729566f30e783af6f31be42ea	Initial Version
2	2 October 2023	7e234fdb16515cd0495e9ace0782c1fb946e9314	After Intermediate Report

For the solidity smart contracts, the compiler version `0.8.17` was chosen. For the purposes of this review we assume the smart contracts will be deployed on Ethereum and executed in the Ethereum EVM Version London. For deployment on other chains/L2s the EVM compatibility must be ensured.

The following files in the folder `src` were in the scope of this review:

- AccessController.sol
- Blacklist.sol
- KIBTAggregator.sol
- KUMABondToken.sol
- KYCToken.sol
- MCAGAggregator.sol

2.1.1 Excluded from scope

Any file not listed above is excluded from the scope.

2.2 System Overview

Mimo Capital AG, a regulated entity, issues ERC-721 compliant NFTs called KUMABondTokens. These tokens represent and are backed by real world bonds. The following smart contracts of mcag-contracts implement the required functionality:

KUMABondToken:

Implements the ERC-721 compliant NFT token issued to the users.

Each token is unique, identified by it's `id` which is tied to the bond the NFT represents. The core information of a bond consists of:

- `cusip`: Bond CUISP number
- `isin`: Bond ISIN number

- `currency`: Currency of the bond
- `term`: Lifetime of the bond in seconds
- `issuance`: Bond issuance date timestamp in seconds
- `maturity`: Annual interest rate paid on the bond - rate per second
- `principal`: Bond face value
- `issuer`: Bond issuer
- `riskCategory`: Unique risk category identifier

The token is pausable, while the token is paused no token can be transferred, issued or redeemed. Furthermore, it implements a blacklist functionality: Blacklisted addresses cannot interact directly with or hold the bond token.

Notably `approve()` allows not only the owner of the NFT to give an approval but also `msg.sender` who are `approvedForAll` on behalf of the NFT's owner.

Admin functionality allows to update the metadata URI, the TNC (terms and conditions), the TNC for a token or the value for max coupon.

KYCToken:

An ERC-721 NFT token issued to a whitelisted address. Can be revoked (burned). All functionality for transferability has been disabled.

AccessController:

Central contract for the role-based access control within the system. Allows the `Default_admin_role` to grant/revoke roles to addresses. Several different roles are present within the system.

Blacklist:

This contract holds a blacklist of addresses. The KUMABondToken uses this blacklist to prevent these addresses to interact with the token.

MCAGAggregator:

Price feed for the central bank rate with a precision of 27 decimals. Implements the well-known Chainlink interface. The price is updated manually by a trusted transmitter role which sets the new value. This function features a few sanity checks which are intended to prevent accidental updates with wrong values.

KIBTAggregator:

Price feed for the KIBToken. KIBTokens are ERC-20 tokens issued by the Kuma-Protocol in exchange for KUMABond NFTs. This price feed serves as an oracle for the MCAGRateFeed contract of the Kuma-Protocol. Similarly to the MCAGAggregator, this pricefeed is updated manually by a trusted transmitter role and implements the well known Chainlink interface.

2.2.1 Trust Model and Roles:

Untrusted roles: Users, NFT Holders. Must not be blacklisted.

Trusted roles:

- `DEFAULT_ADMIN_ROLE`: Fully trusted. Manages the roles: grants / revokes roles to addresses. If compromised, the whole system is compromised.
- `MINT_ROLE`: Can mint KUMABond and KYC tokens.
- `BURN_ROLE`: Can burn KUMABond and KYC tokens.
- `BLACKLIST_ROLE`: Manages the blacklist contract, can add and remove addresses to the blacklist.
- `PAUSE_ROLE`: Can pause the pausable token contracts.

- `UNPAUSE_ROLE`: Can unpause the pausable token contracts.
- `TRANSMITTER_ROLE`: Can update the Price feeds: MCAGAggregator and KIBTAggregator.
- `MANAGER_ROLE`: Can set the `MaxAnswer` and `VolatilityThreshold` in the Aggregator contracts.

Furthermore the following roles can set the respective parameters: `SET_URI_ROLE`, `SET_TNC_ROLE` and `SET_MAX_COUPON_ROLE`.

3 Limitations and use of report

Security assessments cannot uncover all existing vulnerabilities; even an assessment in which no vulnerabilities are found is not a guarantee of a secure system. However, code assessments enable the discovery of vulnerabilities that were overlooked during development and areas where additional security measures are necessary. In most cases, applications are either fully protected against a certain type of attack, or they are completely unprotected against it. Some of the issues may affect the entire application, while some lack protection only in certain areas. This is why we carry out a source code assessment aimed at determining all locations that need to be fixed. Within the customer-determined time frame, ChainSecurity has performed an assessment in order to discover as many vulnerabilities as possible.

The focus of our assessment was limited to the code parts defined in the engagement letter. We assessed whether the project follows the provided specifications. These assessments are based on the provided threat model and trust assumptions. We draw attention to the fact that due to inherent limitations in any software development process and software product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which itself cannot be free from any error or failures. These preconditions can have an impact on the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure which adds further inherent risks as we rely on the correct execution of the included third-party technology stack itself. Report readers should also take into account that over the life cycle of any software, changes to the product itself or to the environment in which it is operated can have an impact leading to operational behaviors other than those initially determined in the business specification.

4 Terminology

For the purpose of this assessment, we adopt the following terminology. To classify the severity of our findings, we determine the likelihood and impact (according to the CVSS risk rating methodology).

- *Likelihood* represents the likelihood of a finding to be triggered or exploited in practice
- *Impact* specifies the technical and business-related consequences of a finding
- *Severity* is derived based on the likelihood and the impact

We categorize the findings into four distinct categories, depending on their severity. These severities are derived from the likelihood and the impact using the following table, following a standard risk assessment procedure.

Likelihood	Impact		
	High	Medium	Low
High	Critical	High	Medium
Medium	High	Medium	Low
Low	Medium	Low	Low

As seen in the table above, findings that have both a high likelihood and a high impact are classified as critical. Intuitively, such findings are likely to be triggered and cause significant disruption. Overall, the severity correlates with the associated risk. However, every finding's risk should always be closely checked, regardless of severity.

5 Findings

In this section, we describe any open findings. Findings that have been resolved have been moved to the Resolved Findings section. The findings are split into these different categories:

- Correctness: Mismatches between specification and implementation

Below we provide a numerical overview of the identified findings, split up by their severity.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	0

6 Resolved Findings

Here, we list findings that have been resolved during the course of the engagement. Their categories are explained in the Findings section.

Below we provide a numerical overview of the identified findings, split up by their severity.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	1

- setMaxAnswer Missing Sanity Check Code Corrected

6.1 `setMaxAnswer` Missing Sanity Check

Correctness Low Version 1 Code Corrected

CS-MCAG-004

The function `MCAGAggregator.setMaxAnswer` simply sets the storage variable `_maxAnswer` to the input argument. As the input argument is defined as signed-integer, mistakenly setting `_maxAnswer` to a negative value, blocks any further calls with positive arguments to `MCAGAggregator.transmit()`.

Code corrected:

Checks to prevent setting negative values have been added to `setMaxAnswer()` and in the constructor:

```
if (maxAnswer_ < 0) {
    revert Errors.MAX_ANSWER_TOO_LOW(maxAnswer_, MIN_MAX_ANSWER);
}
```

and `MCAGAggregator.setMaxAnswer()`:

```
if (newMaxAnswer < 0) {
    revert Errors.MAX_ANSWER_TOO_LOW(newMaxAnswer, MIN_MAX_ANSWER);
}
```

6.2 Mcag accessController Uses Undocumented Roles

Informational Version 1 Specification Changed

CS-MCAG-002

The following two roles are used in `AccessController` of mcag, but not documented in the README:

1. `MCAG_SET_TNC_ROLE`
2. `MCAG_SET_MAX_COUPON_ROLE`

Specification changed:

Mimo Capital AG has added the definition of the two aforementioned roles to the README, as well as functions callable by these two roles.

6.3 Unused Constant `MIN_TERM`

Informational **Version 1** **Code Corrected**

CS-MCAG-003

The defined constant `MIN_TERM` in the contract `KIBTAggregator` is defined but not used.

Code corrected:

Mimo Capital AG has removed the definition of this unused constant.

7 Informational

We utilize this section to point out informational findings that are less severe than issues. These informational issues allow us to point out more theoretical findings. Their explanation hopefully improves the overall understanding of the project's security. Furthermore, we point out findings which are unrelated to security.

7.1 Define Relevant Fields in the Events as Indexed

Informational **Version 1**

CS-MCAG-001

Although some event fields are already defined as indexed, it might make sense to define further ones as indexed as well. Defining event fields as indexed makes searching for specific addresses/values easier.

8 Notes

We leverage this section to highlight further findings that are not necessarily issues. The mentioned topics serve to clarify or support the report, but do not require an immediate modification inside the project. Instead, they should raise awareness in order to improve the overall understanding.

8.1 KIB Price Determination & Volatility

Note Version 1

It may be obvious that one KIB token should be priced at one unit of the underlying principal token. However due to low liquidity, constraints around the redemption of KIB tokens (accruing sufficient KIB tokens, buying the bond of the protocol, off-chain redemption) the price may fluctuate. Furthermore, low liquidity in e.g. Uniswap pools results in notable price volatility / slippage when trading as well as potential price manipulation.

KIBTAggregator serves as a price feed for KIB tokens. This pricefeed returns a value that is manually updated. Its price might deviate from the tokens actual valuation across the DeFi system and must be used with care.

8.2 Updating Terms and Conditions

Note Version 1

`KUMABondToken` allows to update the terms and conditions URL for an already existing Bond NFT. Although being callable only by holder of `MCAG_SET_TNC_ROLE` role, it might bring legal complexities, as users buy a bond token given the terms and conditions during the purchase. Changing terms and conditions for a sold bond token should be treated with extra caution.