



ANASTASIA LABS

Security Audit Report

Date: July 3, 2024
Project: Bond Dex
Version 1.0

Contents

Disclosure	1
Disclaimer and Scope	2
Assessment overview	3
Assessment components	4
Executive summary	5
Code base	6
Repository	6
Commit	6
Files audited	6
Severity Classification	8
Finding severity ratings	9
Findings	10
ID-301 Incorrect datum check in limit bid	11
ID-201 Value size spam in limit ask	12
ID-202 Value size spam in limit bid	13
ID-203 Value size spam in making ask	14
ID-204 Value size spam in making bid	15
ID-101 Potentially locked exchange fee UTxOs	16

Disclosure

This document contains proprietary information belonging to Anastasia Labs. Duplication, redistribution, or use, in whole or in part, in any form, requires explicit consent from Anastasia Labs.

Nonetheless, both the customer Danogo and Anastasia Labs are authorized to share this document with the public to demonstrate security compliance and transparency regarding the outcomes of the Protocol.

Disclaimer and Scope

A code review represents a snapshot in time, and the findings and recommendations presented in this report reflect the information gathered during the assessment period. It is important to note that any modifications made outside of this timeframe will not be captured in this report.

While diligent efforts have been made to uncover potential vulnerabilities, it is essential to recognize that this assessment may not uncover all potential security issues in the protocol.

It is imperative to understand that the findings and recommendations provided in this audit report should not be construed as investment advice.

Furthermore, it is strongly recommended that projects consider undergoing multiple independent audits and/or participating in bug bounty programs to increase their protocol security.

Please be aware that the scope of this security audit does not extend to the compiler layer, such as the UPLC code generated by the compiler or any areas beyond the audited code.

The scope of the audit did not include additional creation of unit testing or property-based testing of the contracts.

Assessment overview

From May 16th, 2024 to June 17th, 2024, Danogo engaged Anastasia Labs to evaluate and conduct a security assessment of its Bond Dex protocol. All code revision was performed following industry best practices.

Phases of code auditing activities include the following:

- Planning – Customer goals are gathered.
- Discovery – Perform code review to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through testing and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities.

The engineering team has also conducted a comprehensive review of protocol optimization strategies.

Each issue was logged and labeled with its corresponding severity level, making it easier for our audit team to manage and tackle each vulnerability.

Assessment components

Manual revision

Our manual code auditing is focused on a wide range of attack vectors, including but not limited to.

- UTXO Value Size Spam (Token Dust Attack)
- Large Datum or Unbounded Protocol Datum
- EUTXO Concurrency DoS
- Unauthorized Data modification
- Multisig PK Attack
- Infinite Mint
- Incorrect Parameterized Scripts
- Other Redeemer
- Other Token Name
- Arbitrary UTXO Datum
- Unbounded protocol value
- Foreign UTXO tokens
- Double or Multiple satisfaction
- Locked Ada
- Locked non Ada values
- Missing UTXO authentication
- UTXO contention

Executive summary

- Danogo Bond Dex is a decentralized exchange platform on the Cardano blockchain, designed to address the absence of a peer-to-peer Bond Trading platform. Danogo Bond Dex offers a flexible solution, allowing bondholders who wish to liquidate their bonds before maturity to find buyers seeking higher yields than those from normal staking rewards.
- When a user buys or sells a bond, the Danogo Bond Dex platform automatically generates a bid or list order with a favorable yield for the owner of the automated order. The platform also facilitates the matching of Liquidity Offers and Borrow Requests to enhance liquidity for users. When a user's Borrow Request is created, these requests are automatically matched with existing Liquidity Offers if the yield requirements are met, and vice versa.
- The primary purpose of Danogo Bond Dex is to facilitate permissionless bond trading within a secure and transparent environment. By integrating open-source smart contracts as a core competency, Danogo Bond Dex ensures security, transparency, and efficiency in bond trading. Our platform focuses on capital protection and low-risk investments for risk-averse and conservative investors seeking stable returns in a secure environment without the volatility of high-risk asset classes.

Code base

Repository

<https://github.com/Danogo2023/bond-dex>

Commit

f35e47368e18a94bf8d9164899647cb120ac6260

Files audited






SHA256 Checksum	Files
255712a2577df54e88e5054b974465970e0369cf07873a17bfa2f3c4dd34d13a	limit-ask/validators/bond_limit.ak
3582a9b2ebb3f00fe75470977db265ac7acc213e8b378094c73f5a460a6a73cf	limit-ask/lib/bond/limit_ask/buy.ak
e89169ce9c49884eda2e14c71fb57e40438ad87250353fc9deae061aeb2859d6	limit-ask/lib/bond/limit_ask/utls.ak
9e3ae2bdf88bc881f5b8d727f75ead784bc2813d93c0cd5295cc8dba0524b464	limit-bid/validators/bond_limit.ak
362da931f170812f41c0fbaa48245491fb2ee14f74d132358ab448d85b46a4b	limit-bid/lib/bond/limit_bid/sell.ak
1cd9556a5998cf1c62c7b5472e9b915dcfa22cba5439d6c6f1e1dab9f2075a49	limit-bid/lib/bond/limit_bid/types.ak
e0ac62cb9ea7d27b76b65ad2b42d04600dcbae4b773243c75758f5aff636723	limit-bid/lib/bond/limit_bid/utls.ak
5f05eb82599ece7f7aa1a0be7235cdf46a8647b92db0f9e3facc6bd44aa68232	limit-bid/lib/bond/limit_bid/withdraw.ak
5185b8b9e820133334d364299af99991c8cb9db024d8173f3608b5824f8e5cf4	making-ask/validators/bond_making.ak
16cf4dbd6f3f0732ca90047bea3d289c09d5f82cab15d8018f0449f5e1ad5fb9	making-ask/lib/bond/making_ask/buy.ak
096cd6062bd9004dfcf476dc53624610e648ae6ebf61dee67c2c000ec07979a9	making-ask/lib/bond/making_ask/utls.ak
62f49d7d310880fca2fe660a5cb715ab9654c934da0f98d1a6dc1044dc664a2c	making-bid/validators/bond_making.ak
e3402c95fd325de9b3586cb21692b68f0104a11a5074e386e0837a26522b08a5	making-bid/lib/bond/making_bid/sell.ak
24c532f75067a1d2c67f543cf9da41c333fcda7170839fa8e515533e79c10d07	making-bid/lib/bond/making_bid/utls.ak

Severity Classification

- **Critical:** This vulnerability has the potential to result in significant financial losses to the protocol. They often enable attackers to directly steal assets from contracts or users, or permanently lock funds within the contract.
- **Major:** Can lead to damage to the user or protocol, although the impact may be restricted to specific functionalities or temporal control. Attackers exploiting major vulnerabilities may cause harm or disrupt certain aspects of the protocol.
- **Medium:** May not directly result in financial losses, but they can temporarily impair the protocol's functionality. Examples include susceptibility to front-running attacks, which can undermine the integrity of transactions.
- **Minor:** Minor vulnerabilities do not typically result in financial losses or significant harm to users or the protocol. The attack vector may be inconsequential or the attacker's incentive to exploit it may be minimal.
- **Informational:** These findings do not pose immediate financial risks. These may include protocol optimizations, code style recommendations, alignment with naming conventions, overall contract design suggestions, and documentation discrepancies between the code and protocol specifications.


Finding severity ratings

The following table defines levels of severity and score range that are used throughout the document to assess vulnerability and risk impact.

	Level	Severity	Findings
	5	Critical	0
	4	Major	0
	3	Medium	1
	2	Minor	4
	1	Informational	1

Findings

ID-301 Incorrect datum check in limit bid

	Level	Severity	Status
	3	Medium	Resolved

Description

The spending validator of the limit-bid UTxO uses the utility function `bid_limit_multi_valid` from the `daken` library to compare the datums of the old and new limit bid UTxOs in a partial fulfillment transaction. This function however does not compare the `bond_types` fields correctly. This can lead to a scenario, where an attacker, while creating a partial fulfill transaction, can modify the `bond_types` field in the residual (output) limit bid UTxO without the consent of the original creator.

Root cause

A single evaluation of the Aiken function `list.difference` by itself is not enough to check that two lists are the same. For example:

```
list.difference([2], [1, 2]) == []
```

Recommendation

It is recommended to modify the implementation of the `bid_limit_multi_valid` function, so it evaluates the difference function both ways:

```
pub fn bid_limit_multi_valid(left: BidLimitMulti, right: BidLimitMulti)
  -> Bool {
  and {
    ...
    (list.difference(left.bond_types, right.bond_types) == [])?,
    (list.difference(right.bond_types, left.bond_types) == [])?,
  }
}
```

Resolution

Resolved in commit `725c607d32d15c815a6901bac37e5a474ba4b984`.

ID-201 Value size spam in limit ask

	Level	Severity	Status
	2	Minor	Resolved

Description

The spending validator of the limit-ask UTxO does not check the size (number of tokens) of the value in the resulting limit-ask UTxO for the partial buy case. This enables an attacker to construct a transaction that purchases a single bond and puts the remaining bonds in a "remainder" limit-ask UTxO along with a huge number of other tokens. This can render the resulting UTxO unspendable in any consequent transaction.

Recommendation

It is recommended to check that apart from the expected bond tokens and Ada no other value is stored on the resulting UTxO for the partial fulfillment of any limit-ask request.

Resolution

Resolved in commit 725c607d32d15c815a6901bac37e5a474ba4b984.

ID-202 Value size spam in limit bid

	Level	Severity	Status
	2	Minor	Resolved

Description

The spending validator of the limit-bid UTxO does not check the size (number of tokens) of the value in the resulting limit-bid UTxO for the partial buy case. This enables an attacker to construct a transaction that purchases a single bond and puts the remaining bonds in a "remainder" limit-bid UTxO along with a huge number of other tokens. This can render the resulting UTxO unspendable in any consequent transaction.

Recommendation

It is recommended to check that apart from the expected bond tokens and Ada no other value is stored on the resulting UTxO for the partial fulfillment of any limit-bid request.

Resolution

Resolved in commit 725c607d32d15c815a6901bac37e5a474ba4b984.

ID-203 Value size spam in making ask

	Level	Severity	Status
	2	Minor	Resolved

Description

The spending validator of the making-ask UTxO does not check the size (number of tokens) of the value in the resulting making-ask UTxO for the partial buy case. This enables an attacker to construct a transaction that purchases a single bond and puts the remaining bonds in a "remainder" making-ask UTxO along with a huge number of other tokens. This can render the resulting UTxO unspendable in any consequent transaction.

Recommendation

It is recommended to check that apart from the expected bond tokens and Ada no other value is stored on the resulting UTxO for the partial fulfillment of any making-ask request.

Resolution

Resolved in commit 8d161f3c80b672b6e61a5e13aae5bd0b5673b427.

ID-204 Value size spam in making bid

	Level	Severity	Status
	2	Minor	Resolved

Description

The spending validator of the making-bid UTxO does not check the size (number of tokens) of the value in the resulting making-bid UTxO for the partial buy case. This enables an attacker to construct a transaction that purchases a single bond and puts the remaining bonds in a "remainder" making-bid UTxO along with a huge number of other tokens. This can render the resulting UTxO unspendable in any consequent transaction.

Recommendation

It is recommended to check that apart from the expected bond tokens and Ada no other value is stored on the resulting UTxO for the partial fulfillment of any making-bid request.

Resolution

Resolved in commit 8d161f3c80b672b6e61a5e13aae5bd0b5673b427.

ID-101 Potentially locked exchange fee UTxOs

	Level	Severity	Status
	1	Minor	Resolved

Description

The limit-ask, limit-bid, making-ask and making-bid validators allows spending limit-ask UTxOs for a full or partial buy transaction. The validator checks that the seller receives a fair price for the bonds and the Danogo exchange receives the prescribed fees. The datum of the fee output UTxOs, where the exchange fee is collected however is not checked, hence a malicious actor can lock these funds in case the fee address is a script address. While this action does not allow stealing funds directly, if the exchange fee address is set to a script address, those funds can be lost.

Recommendation

It is recommended to check the datum of the fee UTxOs to be the void datum, or check that the fee address is a verification key address.

Resolution

Resolved in commit `bc5211540bebeb6cf74d1f8242ce274e86a66478`.