CANTINA

Informal Systems UI vCISO

Security Review

Cantina Managed review by:
Gajinder Singh, Lead Security Researcher

May 28, 2024

Contents

1 Introduction

1.1 About Cantina

Cantina is a security services marketplace that connects top security researchers and solutions with clients. Learn more at cantina.xyz

1.2 Disclaimer

Cantina Managed provides a detailed evaluation of the security posture of the code at a particular moment based on the information available at the time of the review. While Cantina Managed endeavors to identify and disclose all potential security issues, it cannot guarantee that every vulnerability will be detected or that the code will be entirely secure against all possible attacks. The assessment is conducted based on the specific commit and version of the code provided. Any subsequent modifications to the code may introduce new vulnerabilities that were absent during the initial review. Therefore, any changes made to the code require a new security review to ensure that the code remains secure. Please be advised that the Cantina Managed security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.3 Risk assessment

Severity	**Description**
Critical	*Must* fix as soon as possible (if already deployed).
High	Leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
Medium	Global losses <10% or losses to only a subset of users, but still unacceptable.
Low	Losses will be annoying but bearable. Applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.
Gas Optimization	Suggestions around gas saving practices.
Informational	Suggestions around best practices or readability.

1.3.1 Severity Classification

The severity of security issues found during the security review is categorized based on the above table. Critical findings have a high likelihood of being exploited and must be addressed immediately. High findings are almost certain to occur, easy to perform, or not easy but highly incentivized thus must be fixed as soon as possible.

Medium findings are conditionally possible or incentivized but are still relatively likely to occur and should be addressed. Low findings a rare combination of circumstances to exploit, or offer little to no incentive to exploit but are recommended to be addressed.

Lastly, some findings might represent objective improvements that should be addressed but do not impact the project's overall security (Gas and Informational findings).

2 Security Review Summary

Informal Systems offers reliable and secure validator nodes, IBC relayers, and other interchain services in the Cosmos ecosystem and beyond.

From Mar 11th to Mar 29th the Cantina team conducted a review of informal-systems-staking-ui on commit hash 24ff2fcc. The team identified a total of **15** issues in the following risk categories:

- Critical Risk: 1
- High Risk: 0
- Medium Risk: 3
- Low Risk: 6
- Gas Optimizations: 0
- Informational: 5

3 Findings

3.1 Critical Risk

3.1.1 Incorrect withdrawal credentials check

Severity: Critical Risk

Context: App.js#L295-L297

Description: The last 20 bytes (40 characters) of the credentials are matched, but the leading byte (first two chars) and not validated of type `0x01`, otherwise one is sending the deposit with the incorrect type which won't be interpreted as the `address` but some random `bls credentials`. This will lead to loss of funds.

Recommendation: Validate the full credential as `0100....<the withdrawal address`.

Informal Systems: Loaded deposit objects are validated for presence and proper formatting of each required property in BoxForTransactionDetails.tsx#L54 at 472d8d0b. Added a further fix in commit ee6b6bad.

Cantina Managed: Resolved.

3.2 Medium Risk

3.2.1 Use of unvalidated pubkeys to form the URL

Severity: Medium Risk

Context: Pubkeys.js#L16

Description: A malicious or hacked web3 provider could send a response which has injected scripts/`URL`s and using it to build web links for users to click, leading to unsafe execution.

Recommendation: Validate the input file.

Informal Systems: Pubkeys (and other fields) are now validated before being composed into the request URL. See BoxForTransactionDetails.tsx#L103 at 93c09afa.

Cantina Managed: Verified as resolved.

3.2.2 Use of unvalidated `txRespose` to build URL

Severity: Medium Risk

Context: Pubkeys.js#L16

Description: A malicious or hacked web3 provider could send a response which has injected scripts/URLs and using it to build web links for user to click and lead to unsafe execution.

Recommendation: Validate the `txResponse`.

Informal Systems: Response is validated before building links in BoxForTransaction.tsx#L162 at 472d8d0b.

Cantina Managed: Verified as resolved.

3.2.3 Dangerous requests could be executed because of missing file validations

Severity: Medium Risk

Context: App.js#L330

Description: Since file fields are not individually validated, a malicious operator could provide user with a file having dangerous data to make dangerous requests.

Recommendation: Validate the file fields individually in `isValidJson`.

Informal Systems: File is now thoroughly validated in BoxForTransactionDetails.tsx#L54 at 472d8d0b.

Cantina Managed: Verified.

3.3 Low Risk

3.3.1 Unvalidated `connectedAccountAddress` is used to form a url

Severity: Low Risk

Context: BoxForNetworkDetails.tsx#L115

Description: `connectedAccountAddress` is directly assigned without doing hex validation.

Recommendation: Even though `wagmi` seems to be returning validated account address, it would be better to use the `cleanHex` utility before assigning to the `connectedAccountAddress`.

Informal Systems: Addressed in commit 8a067659.

Cantina Managed: Verified as resolved.

3.3.2 Error messgae dumps `Object [object]` but a proper error string

Severity: Low Risk

Context: App.js#L759

Description: Error object is directly rendered to the UI, potentially causing unintended consequences. Error object could have unsanitized data with some injected scripts or clickable URLs.

Recommendation: Only display safe string error messages.

Informal Systems: Errors are now almost entirely hardcoded strings, but caught errors are all converted to `string` before rendering too. See for example BoxForTransaction.tsx#L164 at 21727850.

Cantina Managed: Verified as resolved.

3.3.3 Opening Metamask on page load without user action

Severity: Low Risk

Context: App.js#L45

Description: When the user page loads, it pops Metamask for access. This happens without explicit user action and is bad practice which could force user to reveal accounts without even verifying the page and its authenticity certificate.

Recommendation: Only send request on user click for connect.

Informal Systems: Accounts are not requested until `Connect Wallet` is clicked. See AppHeader.tsx#L41 at 472d8d0b.

Cantina Managed: Verified as resolved.

3.3.4 Invalid deposits could be signed because of missing signature validation

Severity: Low Risk

Context: App.js#L232

Description: The deposit signature is not verified before sending it to the extension for signing and submission. The consequences of this (if not loss of funds) are dependent on the smart contract executing the transaction.

Recommendation: Validate the deposit signature on app or server side.

Informal Systems: A step has been added to first validate user's identity before making the transaction in BoxForTransaction.tsx#L58 at 472d8d0b.

Cantina Managed: This validates that the transaction would be sent to the correct address, however it doesn't validate that the deposit file is valid (valid signature from the pubkey). However if the contract is validating this this issue can be marked resolved (so try uploading an invalid signature file and validate contract rejecting).

Informal Systems: Confirmed the chain wont allow any manipulations of signatures (I changed the signature for a `deposit_data.json` with a valid signature from a different deposit file and its rejected before being able to sign).

Cantina Managed: Resolved.

3.3.5 Garbage/arbitrary data could be signed because of non validated json file

Severity: Low Risk

Context: App.js#L234-L260

Description: Since individual fields of the file are not validated or sanitized, the transaction data send for signing could be arbitrary or garbage. This could have unintended consequences based on the smart contract doing batch processing.

Recommendation: Validate all fields for valid hex (`[0-9, a-f]` and length.

Informal Systems: File is now thoroughly validated in BoxForTransactionDetails.tsx#L54 at 472d8d0b.

Cantina Managed: Verified.

3.3.6 Usage of unsafe `useRef`

Severity: Low Risk

Context: App.js#L694

Description: A reference is used to directly access the dom node for file input. Direct dom manipulation is considered unsafe.

Since its only being used to read the input file target here, it doesn't actually hurt the security of the UI. However, one should avoid direct dom referencing and usage.

Recommendation: File target could also be extracted from the click event, avoiding thus the usage of `useRef` as its an unhealthy coding practice which could devolve into injecting usage html to dom from an unvalidated or user controlled input.

Informal Systems: `useRef` is avoided in favor of event-handling props, in YourFile.tsx#L208 at 472d8d0b.

Cantina Managed: Verified.

3.4 Informational

3.4.1 File data validty check should be stricter

Severity: Informational

Context: BoxForLoadYourFile.tsx#L74

Description: Invalid and garbage data could be assigned to internal state.

Recommendation: File should check for fields validity directly on load and shouldn't even be assigned to internal state. This could also simplify the validation checks later, making the check for public key already submitted easy to follow.

Informal Systems: Acknowledged.

Cantina Managed: Acknowledged.

3.4.2 `JSON.parse` should be in `try`/`catch` with error handling

Severity: Informational

Context: BoxForLoadYourFile.tsx#L57

Description: There is an unhandled error as the file could be an invalid `json`. Although the error is not a security risk, it can cause the application to stop functioning.

Recommendation: Handle the error in try catch.

Informal Systems: Fixed in commit 16af8dc3.

Cantina Managed: Verified as resolved.

3.4.3 `canSendTransaction` should be stricker

Severity: Informational

Context: BoxForTransaction.tsx#L55

Description: UI enables a transaction if even there is one valid deposit.

Recommendation: UI should enable transactions if there are no invalid deposits. Since deposits are batched, any error in one desposit should sort of cast doubt on the entire batch. For example: if one of deposit was previously submitted and has shown up as error, other deposits might not have yet showed up because of whatever reasons, like latency on processing of data by the API provider.

Informal Systems: Addressed in commit bfb3c7b3.

Cantina Managed: Verified as resolved.

3.4.4 Excluded keys check is not precise and might lead to resubmission of a new transaction

Severity: Informational

Context: App.js#L330

Description: Excluded keys are deduced on the basis of the an API call to a beacon API, however this might not be totally accurate as the beacon API URL might be not up to the head or the transaction might not have been included in canonical chain. This could lead to temporary locking of funds even if excess deposits get withdrawn out by the beacon chain eventually.

Recommendation: One needs to keep track of submitted keys in a more precise manner may be server side.

Informal Systems: BoxForTransaction.tsx#L192 at ec3dac11 and BoxForTransactionDetails.tsx#L110 at ec3dac11 start to address this issue but, I will be doing testing with the smart contract to make sure it kicks back funds to the user if a "_double deposi_t" is made.

When we forced a re-submission through a separate contract the double deposit was refunded to the testing wallet so no user funds were lost.

Cantina Managed: This notes an out of gas failure.

Informal Systems: See overview and deposits. Successfully double deposited through this contract and funds are being return on next withdrawal which confirms no users funds would be lost which is the main concern. I will make sure to put more explanation in the UI about what happens with double deposit.

Cantina Managed: The deposits are confirmed withdrawn on double deposits . Please note that: in upcoming Electra upgrade one would be able to deposit > 32 ETH all the way to 2048 ETH, and earn rewards proportionately. But even that is withdrawable.

3.4.5 Use `const` declarations for url prefixes/network/contract instead of react state

Severity: Informational

Context: App.js#L25-L38

Description: The URL and other other constants are used to display the data in UI or form clickable links. If the URL has user input data, that needs to be validated. Although this isn't the case here, but for clarity purposes constants should be used when possible.

Recommendation: Move some of the config values to constants instead of react state.

Informal Systems: Constants have been consolidated in single `constants` module. See constants.ts#L1 at 472d8d0b.

Cantina Managed: Verified as addressed.