Stacking DAO Audit

November 2023

By CoinFabrik

v202307

Executive Summary

CoinFabrik was asked to audit the contracts for the Stacking DAO project.

During this audit we found three critical issues and several minor issues. Also, an enhancement was proposed.

All the issues and enhancements were resolved by the development team.

Scope

The audited files are from the git repository located at https://github.com/StackingDAO/StackingDAO. The audit is based on the commit `3a799d246991fc1ea5652d4c587b81a04d8f59eb`. Fixes were checked on commit `59a9ccf79bd0d182161f5ade2e246d03b47af4b5`.

The scope for this audit includes and is limited to the following files:

- `clarity/contracts/commission-trait-v1.clar`: Trait for commission contracts.
- `clarity/contracts/commission-v1.clar`: Contract for retaining part of the rewards for the protocol.
- `clarity/contracts/core-v1.clar`: Contract for minting stSTX in exchange for STX and for depositing staking rewards.
- `clarity/contracts/dao.clar`: Registry contract for whitelisting protocol contracts and pausing the system.
- `clarity/contracts/reserve-trait-v1.clar`: Trait for reserve contracts.
- `clarity/contracts/reserve-v1.clar`: Contract for holding STX and keeping track of the staked amount.
- `clarity/contracts/stacker-1.clar`: PoX-staker contract.
- `clarity/contracts/staking-trait-v1.clar`: Trait for staking contracts.
- `clarity/contracts/staking-v1.clar`: Contract for staking stDAO to get part of protocol revenue.
- `clarity/contracts/stdao-token.clar`: stDAO token contract.
- `clarity/contracts/strategy-v0.clar`: Contract for managing PoX-staker contracts.
- `clarity/contracts/ststx-token.clar`: stSTX token contract.
- `clarity/contracts/ststx-withdraw-nft.clar`: NFT for withdrawals from the core contract.
- `clarity/contracts/tax-token-trait-v1.clar`: Trait for tax contracts.
- `clarity/contracts/tax-v1.clar`: Contract for collecting taxes from the swaps with stDAO and providing liquidity to a STX/stDAO liquidity pool.

No other files in this repository were audited. Its dependencies are assumed to work according to their documentation. Also, no tests were reviewed for this audit.

Methodology

CoinFabrik was provided with the source code, including automated tests that define the expected behavior, and general documentation about the project. Our auditors spent two weeks auditing the source code provided, which includes understanding the context of use, analyzing the boundaries of the expected behavior of each contract and function, understanding the implementation by the development team (including dependencies beyond the scope to be audited) and identifying possible situations in which the code allows the caller to reach a state that exposes some vulnerability. Without being limited to them, the audit process included the following analyses.

- Arithmetic errors
- Race conditions
- Reentrancy attacks
- Misuse of block timestamps
- Denial of service attacks
- Excessive gas usage
- Missing or misused function qualifiers
- Needlessly complex code and contract interactions
- Poor or nonexistent error handling
- Insufficient validation of the input parameters
- Incorrect handling of cryptographic signatures
- Centralization and upgradeability

After delivering a report with our findings, the development team had the opportunity to comment on every finding and fix the issues they considered convenient. Once fixed and/or commented, our team ran a second review process to verify that the changes to the code effectively solve the issues found and do not unintentionally add new ones. This report includes the final status after the second review.

Findings

In the following table we summarize the security issues we found in this audit. The severity classification criteria and the status meaning are explained below. This table does not include the enhancements we suggest to implement, which are described in a specific section after the security issues.

ID	Title	Severity	Status
CR-01	Blocked Withdrawals and Stolen Rewards	Critical	Resolved
CR-02	Reward Miscalculation	Critical	Resolved
CR-03	Update Codebase for Mainnet	Critical	Resolved
MI-01	Panicking on Possible Error	Minor	Resolved
MI-02	No Limit for the Reward Commission	Minor	Resolved
MI-03	No Limit for the Staking Percentage	Minor	Resolved
MI-04	Any Whitelisted Principal Can Disable Others	Minor	Resolved

Severity Classification

Security risks are classified as follows:

- **Critical:** These are issues that we manage to exploit. They compromise the system seriously. Blocking bugs are also included in this category. They must be fixed **immediately**.
- **Medium:** These are potentially exploitable issues. Even though we did not manage to exploit them or their impact is not clear, they might represent a security risk in the near future. We suggest fixing them **as soon as possible**.
- **Minor:** These issues represent problems that are relatively small or difficult to take advantage of, but might be exploited in combination with other issues. These kinds of issues do not block deployments in production environments. They should be taken into account and be fixed **when possible**.

Issues Status

An issue detected by this audit has one of the following statuses:

- **Unresolved**: The issue has not been resolved.

- **Acknowledged**: The issue remains in the code, but is a result of an intentional decision. The reported risk is accepted by the development team.
- **Resolved**: Adjusted program implementation to eliminate the risk.
- **Partially resolved**: Adjusted program implementation to eliminate part of the risk. The other part remains in the code, but is a result of an intentional decision.
- **Mitigated**: Implemented actions to minimize the impact or likelihood of the risk.

Critical Severity Issues

CR-01 Blocked Withdrawals and Stolen Rewards

Location:

- `contracts/core-v1.clar:197, 199, 200, 229, 231`

The withdrawals from the system are performed through the core contract, which burns users' stSTX and returns the corresponding STX. The withdrawals can be blocked by setting `shutdown-withdrawals` to `true` or disabling the core contract from the `contracts` map in `dao.clar`. The administrator is the only user capable of doing it.

In combination with MI-02 and MI-03, a malicious administrator can freeze the funds, set the reward commission to 100% and the staking percentage to 0%, so that all the earned rewards are assigned to himself.

Also, the withdrawals are limited by a threshold that prohibits extracting more than a certain percentage of the total amount of STX staked on a specific cycle. If that limit is reached, users should need to wait for the next cycle to withdraw from their assets. Initially, the threshold is set to 85%, but the administrator can set any new value through `set-withdrawal-treshold()`. Therefore, this value can be set to zero and it is another method for freezing all the funds.

Recommendation

1. Remove these withdrawal restrictions or reduce the incentives by resolving MI-02 and MI-03.
2. Define a minimum value for `set-withdrawal-treshold()` and enforce it.

Status

Resolved. Withdrawal restrictions were removed as well as the threshold.

CR-02 Reward Miscalculation

Location:

- `contracts/staking-v1.clar:219-239`

The `calculate-cumm-reward-per-stake()` function in the `stacking-v1` contract exhibits a security vulnerability. After assigning rewards to each staked token via `add-rewards()`, the `rewards-per-block` variable is not properly deinitialized. Consequently, if the function is called again in the next block without adding new rewards, it will incorrectly return a higher reward per token. This oversight allows for unintended reward inflation, as the system fails to differentiate between rewards from the previous cycle and newly set ones.

Recommendation
Implement proper deinitialization of the `rewards-per-block` variable after each distribution cycle to ensure accurate reward calculations.

Status
Resolved. There is now a specific end block at which the reward distribution stops.

CR-03 Update Codebase for Mainnet

The codebase has many instances of comment with the text “TODO: update for mainnet”, referring to hardcoded addresses and other environment variables.
These instances should be resolved before deploying to mainnet.

Recommendation
Execute planned modifications for mainnet.

Status
Resolved. Mainnet modifications implemented.

Medium Severity Issues

No issues found.

Minor Severity Issues

MI-01 Panicking on Possible Error

Location:

- `contracts/ststx-withdraw-nft.clar:137`
- `contracts/core-v1.clar:233`
- `contracts/reserve-v1:40`

Using `unwrap-panic` results in the transaction being finished because of a runtime error when the provided value is an error or a `none`. The runtime error does not allow the caller to handle that error and act in response. Also, this kind of error does not provide any information about the reason for the reverted transaction to the user.

While that form is a convenient method to unwrap values, it should not be used unless it is seemingly impossible to trigger the panic.

Recommendation
Replace `unwrap-panic` for `unwrap!` when there is a flow which might trigger an error or none value (instances listed under Location).

Status
Resolved. Fixed according to the recommendation.

MI-02 No Limit for the Reward Commission

Location:

- `contracts/core-v1.clar:263, 299`

The commission is the value used to calculate how much from the PoX rewards is kept for the protocol and the stDAO stakers. The greater the commission, the lower the increase on the reserve contract, and consequently the lower the rewards for STX stakers.

The commission is defined by the administrator through `set-commission()` and can be set to any value, even 100%.

Recommendation
Define a maximum value for the commission and enforce it.

Status
Resolved. Fixed according to the recommendation.

MI-03 No Limit for the Staking Percentage

Location:

- `contracts/commission-v1.clar:32, 72`

The staking percentage defines how much from the reward commission is assigned to stDAO stakers, while the rest of the commission is for the protocol.

This percentage is defined by the administrator through `set-staking-percentage()` and can be set to any value, even 0%.

Recommendation
Define a minimum value for the percentage and enforce it.

Status
Resolved. Fixed according to the recommendation.

MI-04 Any Whitelisted Principal Can Disable Others

Location:

- `contracts/dao.clar:78`

System contracts are whitelisted in the `contracts` map. This map is queried by the other contracts to authenticate the caller in restricted functions. Through the function `set-contract-enabled()`, principals can be enabled or disabled.

However, not only system contracts are initially added to this whitelist, but also the deployer of the dao contract. This deployer will act as administrator of the system. But, since the only role is defined in the `contracts` map, any principal in that map can call `set-contract-enabled()`, disabling others and taking control over the system.

In the current state, the dao deployer is the only EOA in the whitelist and the other contracts do not specify an interaction with that function. But if new EOAs are added to the whitelist, this issue can be exploited.

Recommendation

Define a different role for the administrator and restrict the role assignment functions to that user.

Status

Resolved. Admin role implemented and role assignment functions are restricted to that role.

Enhancements

These items do not represent a security risk. They are best practices that we suggest implementing.

ID	Title	Status
EN-01	Remove Duplicated Validation	Implemented

EN-01 Remove Duplicated Validation

Location:

- `contracts/core-v1.clar:164, 168`

In the deposit function, the parameter `reserve-contract` is validated twice, One within the context of the `get-stx-per-ststx()` function, called on line 164, and again on line 168.

Recommendation
Remove the assertion on line 168.

Status
Implemented.

Other Considerations

The considerations stated in this section are not right or wrong. We do not suggest any action to fix them. But we consider that they may be of interest to other stakeholders of the project, including users of the audited contracts, token holders or project investors.

Centralization

The system has an administrator who sets most of the system parameters and can shutdown the system whenever he wants, freezing the funds as described on CR-01. There is no voting mechanism for allowing the stakers participate in the administration process.

Upgrades

Excluding the commission contract, most of the addresses on the system are hardcoded. Therefore, upgrading specific isolated components is not possible, it might require a more extensive upgrade. However, new contracts could be added to the system by whitelisting them on `dao.clar`.

Changelog

- 2023-11-14 – Initial report based on commit `3a799d246991fc1ea5652d4c587b81a04d8f59eb`.
- 2023-12-05 – Reaudit report based on the fixes in commit 59a9ccf79bd0d182161f5ade2e246d03b47af4b5.

Disclaimer: This audit report is not a security warranty, investment advice, or an approval of the Stacking DAO project since CoinFabrik has not reviewed its platform. Moreover, it does not provide a smart contract code faultlessness guarantee.