AlexGo Audit

Stxdx-v1

July 2022

By CoinFabrik

Introduction

CoinFabrik was asked to audit the contracts for the AlexGo project. Stxdx-v1 is an order matching protocol for peer to peer exchange. First we will provide a summary of our discoveries, and then we will show the details of our findings.

Scope

The audited files are from the git repository located at https://github.com/alexgo-io/alex-stxdx. The audit is based on the commit `3c6a9272402dee95aa386c5d8e767207f0ca441e`. Fixes were reviewed on commit `13b80bc64f2f4c7730c4f27fb8a3d67cbdbc3785`.

The audited files are:

- `contracts/stxdx-exchange-zero.clar`: Exchange contract where orders are matched.
- `contracts/stxdx-registry.clar`: Users, assets, order fills and approvals are registered in this contract.
- `contracts/stxdx-sender-proxy.clar`: Proxy for stxdx-exchange-zero.
- `contracts/stxdx-wallet-zero.clar`: Wallet where users store their assets to operate with the exchange.
- `contracts/stxdx-utils.clar`: Utility functions for the system.

The scope of the audit is limited to those files. 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.

Analyses

Without being limited to them, the audit process included the following analyses:

- Arithmetic errors
- Race conditions
- 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

Summary of Findings

We found one medium and one minor issues. No enhancements were proposed.

The minor issue was fixed, while the medium was mitigated.

Security Issues

ID	Title	Severity	Status
ME-01	Insecure Authentication Through tx-sender	Medium	Mitigated
MI-01	Asset Registered Twice With Different ID Numbers	Minor	Resolved

Security Issues Found

Severity Classification

Security risks are classified as follows:

- **Critical:** These are issues that we manage to exploit. They compromise the system seriously. 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 can 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 can have four distinct statuses:

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

- **Acknowledged**: The issue remains in the code, but is a result of an intentional decision.
- **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

No issues found.

Medium Severity Issues

ME-01 Insecure Authentication Through tx-sender

Location:

- `contracts/stxdx-exchange-zero.clar`,
- `contracts/stxdx-registry.clar`,
- `contracts/stxdx-sender-proxy.clar`,
- `contracts/stxdx-wallet-zero.clar`

Global variable `tx-sender` returns the original sender of the current transaction, or if `as-contract` was called to modify the sending context, it returns that contract principal. Using this variable for authentication is not secure. Actors in the system could be targeted for phishing. Thanks to post-conditions, functions which involve asset transfers cannot be called in the attack, so there it is safe to use `tx-sender`.

This issue was found in:

- `stxdx-sender-proxy::is-contract-owner()`,
- `stxdx-exchange-zero::is-contract-owner()`,
- `stxdx-exchange-zero::validate-authorisation()`,
- `stxdx-exchange-zero::approve-order()`,
- `stxdx-registry::set-contract-owner()`,
- `stxdx-registry::is-contract-owner()`,
- `stxdx-registry::register-user()`,
- `stxdx-registry::set-order-approval()`,
- `stxdx-wallet-zero::is-contract-owner()`,
- `stxdx-wallet-zero::is-authorised-approver()`,

- `stxdx-wallet-zero::request-transfer-out()`.

However, in functions like `is-contract-owner()`, `set-contract-owner()`, and others that authenticate the contract owner, the risk is mitigated when a DAO is used as owner.

Recommendation
Prefer `contract-caller` to `tx-sender` for authentication, unless it is specifically required and the risk is considered.

Status
Mitigated. Since the issue is already mitigated for the cases previously described, the development team decided not to follow the recommendation, favoring composability, which would have been compromised with the proposed change.

Minor Severity Issues

MI-01 Asset Registered Twice With Different ID Numbers

Location:

- `contracts/stxdx-registry.clar:47-58`

In `register-asset()`, the new principal to be set might be already registered, and this is not verified at registration. As a consequence, the same asset can be identified with multiple ID numbers, which is not an issue in the current state of the system, but it might cause a bug if the ID were used to differentiate assets.

Recommendation
Verify the principal is not already registered before the call to `register-asset()`. For instance, the last `map-set` could be replaced with:

```
(asserts! (map-insert asset-registry-ids asset asset-id)
err-asset-already-registered)
```

Status
Resolved. Fixed according to the recommendation.

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 for other stakeholders of the project, including users of the audited contracts, owners or project investors.

Centralization

Orders are generated off-chain and order books are hosted off-chain, then exchange liveness relies also on a service provided by a server out of the blockchain.

Specifically, off-chain order generation is out of the scope of this audit, and it is a key component since cryptographic signatures are used. The orders created have a salt value, which is mandatory in order to avoid signature replay attacks, but the salt generation and inclusion is made off-chain and that is not verified on this audit.

Additionally, in `stxdx-wallet-zero::transfer()`, approved exchanges can transfer on behalf of any user, without limitations regarding if the exchange which is moving the assets is the one originally used by the user. Therefore, users should trust that the AlexGo team is not going to perform a malicious exchange.

Changelog

- 2022-07-04 – Initial report based on commit `3c6a9272402dee95aa386c5d8e767207f0ca441e`.
- 2022-07-08 – Final report based on commit `13b80bc64f2f4c7730c4f27fb8a3d67cbdbc3785`.

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