Trust Machines MS Audit

Multisafe

July 2022

By CoinFabrik

Introduction

CoinFabrik was asked to audit the contracts for Trust Machines MS's Multisafe project. This is a shared crypto wallet for managing Stacks (STX) and Bitcoin (BTC). 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 github.com/Trust-Machines/multisafe/. The audit is based on the commit `61c5c327918070205a1b2514a12ee47637025100`. Fixes were checked on commit `53cfded5498fd77ec4341e16cc3674fe1d61908b`.

The audited files are:

- `./contracts/executors/add-owner.clar`: Executor contract to add a new owner for the wallet.
- `./contracts/executors/allow-caller.clar`: Executor contract to add a new authorized caller to the wallet contract.
- `./contracts/executors/magic-bridge-set.clar`: Executor contract to set the magic bridge contract address in the wallet.
- `./contracts/executors/remove-owner.clar`: Executor contract to remove a new owner for the wallet.
- `./contracts/executors/revoke-caller.clar`: Executor contract to revoke an authorized caller to the wallet contract.
- `./contracts/executors/set-threshold.clar`: Executor contract to modify threshold value.
- `./contracts/executors/set-vault-token-per-cycle.clar`: Executor contract to set a new value for token-per-cycle in the vault contract.
- `./contracts/executors/transfer-sip-009.clar`: Executor contract to transfer SIP-009 tokens out of the wallet.
- `./contracts/executors/transfer-sip-010.clar`: Executor contract to transfer SIP-010 tokens out of the wallet.
- `./contracts/executors/transfer-stx.clar`: Executor contract to transfer STX tokens out of the wallet.
- `./contracts/executors/magic-bridge-send.clar`: Executor contract to bridge xBTC tokens out of the wallet to the Bitcoin network.
- `./contracts/helper/ft-none.clar`: Helper SIP-010 token with no functionalities used as a workaround for the non-existent trait-optional type.

- `./contracts/helper/nft-none.clar`: Helper SIP-009 token with no functionalities used as a workaround for the non-existent trait-optional type.
- `./contracts/misc/vault.clar`: An example vault contract for simulation executor.
- `./contracts/misc/magic-bridge.clar`: Mocked magic bridge contract.
- `./contracts/safe.clar`: Multisig wallet. This is the main contract.
- `./contracts/traits.clar`: Traits definitions 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 a critical issue, and a medium issue. Also, several enhancements were proposed.

All the issues were addressed.

Security Issues

ID	Title	Severity	Status
CR-01	Owner Can Execute Anything On Behalf Of The Wallet	Critical	Resolved
ME-01	Insecure Authentication Through tx-sender	Medium	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

CR-01 Owner Can Execute Anything On Behalf Of The Wallet

Location:

- `contracts/safe.clar:341-387`

Functions for Magic Bridge interaction (`mb-initialize-swapper()` and `mb-escrow-swap()`) can be called by any owner, without approvals from the rest of the owners. This is supposed to be used to interact with the bridge contract, but it could be used with any contract which is compliant with the bridge trait.

These functions execute remote code from the contract passed by parameter. This remote call is enclosed in the `as-contract` expression, which changes the `tx-sender` value to the wallet contract. This enables the remote code to act on behalf of the wallet (e.g. transferring its assets out of it).

Therefore, any owner can create a malicious contract compliant to the bridge trait and execute anything on behalf of the wallet, avoiding the transaction and voting system implemented. This could lead to transferring assets out of the wallet.

Recommendation

The bridge address should be a variable whose value is consent by the owners.

Status

Resolved. Fixed according to the recommendation. A new executor contract (magic-bridge-set) was created in order to set a new value to the variable.

Medium Severity Issues

ME-01 Insecure Authentication Through tx-sender

Location:

- `contracts/safe.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.

In `submit()`, `confirm()`, and `revoke()`, `tx-sender` is used to authenticate the owner, but if owners were victims of phishing, malicious transactions could be executed by the wallet.

Recommendation

Consider using contract-caller for authentication, especially when owners are externally owned accounts.

Status

Resolved. Fixed according to recommendation. Also, a registry for authorized caller was implemented in order to whitelist specific addresses in order to allow other multisafe wallets as owners.

Minor Severity Issues

No issues found.

Enhancements

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

Table

ID	Title	Status
EN-01	Reverting With Error Code Instead Of Panicking	Implemented
EN-02	Threshold Initialization And Owner Removing	Implemented
EN-03	Unnecessary Assertion On Threshold Setting	Implemented

Details

EN-01 Reverting With Error Code Instead Of Panicking

Location:

- `contracts/safe.clar:68`

`add-owner-internal()` results in a runtime error if the number of owners surpasses the maximum specified in the code. This is because of the `unwrap-panic` expression. However, a runtime error is not informative and the condition for the error is likely to occur.

Recommendation

Prefer `unwrap!` with an error code instead of `unwrap-panic` unless the condition is extremely unlikely.

Status

Implemented.

EN-02 Threshold Initialization And Owner Removing

Location:

- `contracts/safe.clar`

The threshold variable is initialized in zero, when the setting function constrains the new value in the range from one to twenty. The variable should be initialized with a value compliant with those constraints.

If that value is initialized in a value different from zero, then the assertion made at line 102 is unnecessary, since the next assertion would contain that condition.

Recommendation
Threshold should be initialized in a value contained in the range defined by the setter function, and the assertion in L102 could be removed.

Status
Implemented.

EN-03 Unnecessary Assertion On Threshold Setting

Location:

- `contracts/safe.clar:135-136`

Assertion at line 135 constrains the new value to be at most twenty:

```
(asserts! (<= value u20) ERR-THRESHOLD-OVERFLOW)
```

While this values is also limited to be at most the length of the owners list at line 136:

```
(asserts! (<= value (len (var-get owners))) ERR-THRESHOLD-OVERFLOW-OWNERS)
```

Since the owners list is limited to twenty because of the data type, its length will always be equal or lower than that value. Therefore, the first assertion is already contained in the second one.

Recommendation
Consider removing assertion at line 135 in order to reduce runtime cost.

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

Upgrades

Wallet functionalities can be extended through executor contracts, which need to be executed by the transaction system. Therefore, these extensions need to be voted on by the owners.

Changelog

- 2022-07-15 – Initial report based on commit `61c5c327918070205a1b2514a12ee47637025100`.
- 2022-07-26 – Final report based on commit `53cfded5498fd77ec4341e16cc3674fe1d61908b`.

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