Chin-Program Audit

September 2022

Updated in June 2024

By CoinFabrik

Introduction

CoinFabrik was asked to audit the Chin Solana program for the Fitchin project. 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/42i-co/chin-program.git. The audit is based on the commit `f9bd6340f8e44dd8c799ef4b8e239b5aaea51cbb`.

Fixes were checked on commit `664661f94b6af229d7aeff9b997d5474d1920f51`.

After that, we looked at the differences on commit `c87bb7369ce63ad640587f72022a11d25aba968d` and found no new additional issues. This commit is on the https://github.com/FITCHIN-1/chin-program.git repository. The new repository contains the history of the repository, including the two commits mentioned above.

The audited files, included in the `programs/chin/src` directory, are:

- `data/access_level.rs`: Access level enum.
- `data/multisig/access_grant_instruction.rs`: `AccessGrantInstruction` account struct, used for the corresponding multisig instruction.
- `data/multisig/access_revoke_instruction.rs`: `AccessRevokeInstruction` account struct, used for the corresponding multisig instruction.
- `data/multisig/mod.rs`: Exports all the account structs defined in the module.
- `data/multisig/multisig_modify_instruction.rs`: `MultisigModifyInstruction` account struct, used for the corresponding multisig instruction.
- `data/multisig/treasury_burn_instruction.rs`: `TreasuryBurnInstruction` account struct, used for the corresponding multisig instruction.
- `data/multisig/treasury_transfer_instruction.rs`: `TreasuryTransferInstruction` account struct, used for the corresponding multisig instruction.

- `data/multisig/vesting_add_instruction.rs`: `VestingAddInstruction` account struct, used for the corresponding multisig instruction.
- `data/multisig/vesting_remove_instruction.rs`: `VestingRemoveInstruction` account struct, used for the corresponding multisig instruction.
- `data/multisig/vesting_transfer_instruction.rs`: `VestingTransferInstruction` account struct, used for the corresponding multisig instruction.
- `data/program_data.rs`: `ProgramData` account struct. It contains the state used globally in the program, such as administrators, operators, multisig threshold, etc.
- `data.rs`: Exports for the `data` module.
- `data/vesting_curve.rs`: Data definition for a vesting curve.
- `data/vesting_wallet.rs`: Data definition for a vesting wallet
- `errors.rs`: Error definitions.
- `instructions/access_grant/access_grant_cancel.rs`: Access grant cancel instruction implementation.
- `instructions/access_grant/access_grant_create.rs`: Access grant create instruction implementation.
- `instructions/access_grant/access_grant_execute.rs`: Access grant execute instruction implementation.
- `instructions/access_grant/access_grant_sign.rs`: Access grant sign instruction implementation.
- `instructions/access_grant/mod.rs`: Exports access grant instructions.
- `instructions/access_revoke/access_revoke_cancel.rs`: Access revoke cancel instruction implementation.
- `instructions/access_revoke/access_revoke_create.rs`: Access revoke create instruction implementation.
- `instructions/access_revoke/access_revoke_execute.rs`: Access revoke execute instruction implementation.
- `instructions/access_revoke/access_revoke_sign.rs`: Access revoke sign instruction implementation.
- `instructions/access_revoke/mod.rs`: Export access revoke instructions.
- `instructions/initialize.rs`: Initialize instruction implementation.
- `instructions/multisig_modify/mod.rs`: Exports multisig modify instructions.
- `instructions/multisig_modify/multisig_modify_cancel.rs`: Multisig modify cancel instruction implementation.
- `instructions/multisig_modify/multisig_modify_create.rs`: Multisig modify create instruction implementation.

- `instructions/multisig_modify/multisig_modify_execute.rs`: Multisig modify execute instruction implementation.
- `instructions/multisig_modify/multisig_modify_sign.rs`: Multisig modify sign instruction implementation.
- `instructions/operations_transfer.rs`: Operations transfer instruction implementation.
- `instructions.rs`: Exports all the instructions defined in the `instructions` module.
- `instructions/treasury_burn/mod.rs`: Exports treasury burn instructions.
- `instructions/treasury_burn/treasury_burn_cancel.rs`: Treasury burn cancel instruction implementation.
- `instructions/treasury_burn/treasury_burn_create.rs`: Treasury burn create instruction implementation.
- `instructions/treasury_burn/treasury_burn_execute.rs`: Treasury burn execute instruction implementation.
- `instructions/treasury_burn/treasury_burn_sign.rs`: Treasury burn sign instruction implementation.
- `instructions/treasury_transfer/mod.rs`: Exports treasury transfer instructions.
- `instructions/treasury_transfer/treasury_transfer_cancel.rs`: Treasury transfer cancel instruction implementation.
- `instructions/treasury_transfer/treasury_transfer_create.rs`: Treasury transfer create instruction implementation.
- `instructions/treasury_transfer/treasury_transfer_execute.rs`: Treasury transfer execute instruction implementation.
- `instructions/treasury_transfer/treasury_transfer_sign.rs`: Treasury transfer sign instruction implementation.
- `instructions/vesting_add/mod.rs`: Exports vesting add instructions.
- `instructions/vesting_add/vesting_add_cancel.rs`: Vesting add cancel instruction implementation.
- `instructions/vesting_add/vesting_add_create.rs`: Vesting add create instruction implementation.
- `instructions/vesting_add/vesting_add_execute.rs`: Vesting add execute instruction implementation.
- `instructions/vesting_add/vesting_add_sign.rs`: Vesting add sign instruction implementation.
- `instructions/vesting_claim.rs`: Vesting claim instruction implementation.
- `instructions/vesting_remove/mod.rs`: Exports vesting remove instructions.

- `instructions/vesting_remove/vesting_remove_cancel.rs`: Vesting remove cancel instruction implementation.
- `instructions/vesting_remove/vesting_remove_create.rs`: Vesting remove create instruction implementation.
- `instructions/vesting_remove/vesting_remove_execute.rs`: Vesting remove execute instruction implementation.
- `instructions/vesting_remove/vesting_remove_sign.rs`: Vesting remove sign instruction implementation.
- `instructions/vesting_transfer/mod.rs`: Exports vesting transfer instructions.
- `instructions/vesting_transfer/vesting_transfer_cancel.rs`: Vesting transfer cancel instruction implementation.
- `instructions/vesting_transfer/vesting_transfer_create.rs`: Vesting transfer create instruction implementation.
- `instructions/vesting_transfer/vesting_transfer_execute.rs`: Vesting transfer execute instruction implementation.
- `instructions/vesting_transfer/vesting_transfer_sign.rs`: Vesting transfer sign instruction implementation.
- `lib.rs`: Entry point for the anchor program.
- `metadata.rs`: Reexports `metadata/security_txt.rs` in the `metadata` namespace.
- `metadata/security_txt.rs`: Metadata describing the project.
- `utils.rs`: It contains some constant definitions and the `min_administrators` function.

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.

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 program interactions
- Poor or nonexistent error handling

- Insufficient validation of the input parameters
- Incorrect account validation
- Incorrect handling of cryptographic signatures
- Centralization

Summary of Findings

We found no critical or medium issues. We found several minor issues. Also, several enhancements were proposed.

All minor issues were fixed. One of the proposed enhancements was made.

Security Issues

ID	Title	Severity	Status
MI-01	Administrator Denial of Service	Minor	Resolved
MI-02	Revoked-Administrator Signatures Are Still Valid	Minor	Resolved
MI-03	Missing Funds on Vesting Remove	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

No issues found.

Minor Severity Issues

MI-01 Administrator Denial of Service

Location:

- `instructions/access_grant/access_grant_cancel.rs: 9,22`
- `instructions/access_revoke/access_revoke_cancel.rs: 9,22`
- `instructions/multisig_modify/multisig_modify_cancel.rs: 9,22`
- `instructions/treasury_burn/treasury_burn_cancel.rs: 9,22`
- `instructions/treasury_transfer/treasury_transfer_cancel.rs: 9,22`
- `instructions/vesting_add/vesting_add_cancel.rs: 9,22`
- `instructions/vesting_remove/vesting_remove_cancel.rs: 9,22`
- `instructions/vesting_transfer/vesting_transfer_cancel.rs: 9,22`

A single rogue administrator may significantly disrupt the operation of the program by canceling all the other administrators actions before the execution of all the required instructions to sign the action and execute it.

This issue is exacerbated by the fact that removing an administrator is a multisig action that can be attacked in this manner and that they may even gain funds by collecting the rent of the action account being destroyed on cancellation.

But this issue's severity is lowered by the fact that it is possible to handcraft a transaction with all the instructions required to kick out the rogue administrator. These are the `access_revoke_create`, `access_revoke_sign` and `access_revoke_execute` instructions.

Recommendation

Only the administrator that created the action request should be able to cancel it. Extra care should be given to administrator action requests that were created by the revoked administrator, which should be assigned a new creator or canceled.

Status

Resolved. Only the creator can cancel an administrator action.

MI-02 Revoked-Administrator Signatures Are Still Valid

Location:

- `instructions/access_grant/access_grant_execute.rs: 23`
- `instructions/access_revoke/access_revoke_execute.rs: 23`
- `instructions/multisig_modify/multisig_modify_execute.rs: 23`
- `instructions/treasury_burn/treasury_burn_execute.rs: 29`
- `instructions/treasury_transfer/treasury_transfer_execute.rs: 26`
- `instructions/vesting_add/vesting_add_execute.rs: 25`
- `instructions/vesting_remove/vesting_remove_execute.rs: 25`
- `instructions/vesting_transfer/vesting_transfer_execute.rs: 26`

If an administration action is in the process of signing and one of the administrators who signed the request is revoked by an `access_revoke` action, its signature is still counted as valid for this action.

This may weaken the security of the system, because a fewer number of actual administrators will be required to sign this action in order to execute it.

Recommendation

Check on all `*_execute` instructions that the number of signers that are in the `program_data.administrators` array is enough to do the action and filter out the revoked administrators on all `*_sign` instructions.

Status

Resolved. Each time an administrator action is signed, the previous administrator signatures are controlled and, if the administrator has been revoked, it is removed from the signatures.

MI-03 Missing Funds on Vesting Remove

Location:

- `instructions/vesting_remove/vesting_remove_execute.rs: 38-40`

When an administrator removes a vesting via the `vesting_remove_execute` instruction, the funds that were awarded to the recipient but not yet claimed cannot be transferred to the vesting recipient anymore.

Recommendation

Send the already awarded funds to the vesting recipient in the `vesting_remove_execute` instruction implementation.

Status

Resolved. When removing a vesting, the already awarded funds are sent to the recipient.

Enhancements

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

ID	Title	Status
EN-01	Repetitions in Multisig Implementation	Not implemented
EN-02	Administrator Signature Not Required on Execute	Implemented

Details

EN-01 Repetitions in Multisig Implementation

Location:

- `instructions/access_grant/access_grant_cancel.rs`
- `instructions/access_grant/access_grant_create.rs`
- `instructions/access_grant/access_grant_execute.rs`
- `instructions/access_grant/access_grant_sign.rs`
- `instructions/access_revoke/access_revoke_cancel.rs`
- `instructions/access_revoke/access_revoke_create.rs`

- `instructions/access_revoke/access_revoke_execute.rs`
- `instructions/access_revoke/access_revoke_sign.rs`
- `instructions/multisig_modify/multisig_modify_cancel.rs`
- `instructions/multisig_modify/multisig_modify_create.rs`
- `instructions/multisig_modify/multisig_modify_execute.rs`
- `instructions/multisig_modify/multisig_modify_sign.rs`
- `instructions/treasury_burn/treasury_burn_cancel.rs`
- `instructions/treasury_burn/treasury_burn_create.rs`
- `instructions/treasury_burn/treasury_burn_execute.rs`
- `instructions/treasury_burn/treasury_burn_sign.rs`
- `instructions/treasury_transfer/treasury_transfer_cancel.rs`
- `instructions/treasury_transfer/treasury_transfer_create.rs`
- `instructions/treasury_transfer/treasury_transfer_execute.rs`
- `instructions/treasury_transfer/treasury_transfer_sign.rs`
- `instructions/vesting_add/vesting_add_cancel.rs`
- `instructions/vesting_add/vesting_add_create.rs`
- `instructions/vesting_add/vesting_add_execute.rs`
- `instructions/vesting_add/vesting_add_sign.rs`
- `instructions/vesting_remove/vesting_remove_cancel.rs`
- `instructions/vesting_remove/vesting_remove_create.rs`
- `instructions/vesting_remove/vesting_remove_execute.rs`
- `instructions/vesting_remove/vesting_remove_sign.rs`
- `instructions/vesting_transfer/vesting_transfer_cancel.rs`
- `instructions/vesting_transfer/vesting_transfer_create.rs`
- `instructions/vesting_transfer/vesting_transfer_execute.rs`
- `instructions/vesting_transfer/vesting_transfer_sign.rs`

The multisig mechanism is implemented 8 times in the source code, once for each administrator's action. This makes any error correction or improvement in the schema much more difficult. For example, corrections corresponding to MI-01 Administrator Denial of Service and MI-02 Revoked-Administrator Signatures Are Still Valid need to be implemented 8 times each, with all the risk involved in having to keep in sync 8 copies of the same logic.

Recommendation

We have 3 different recommendations to resolve this problem. Each one has a different trade-off in functionality and simplicity:

1. Use Solana's ability to have multiple signers for each instruction in a transaction and have all the administrator instructions check that all the required signatures are present. Extract the signature-check logic to a

function and invoke this function on each action. If this recommendation is implemented MI-01 Administrator Denial of Service and MI-02 Revoked-Administrator Signatures Are Still Valid would be fixed.
2. Use a different Solana program to handle the multiple-signers logic and make this program the administrator of the system. If this recommendation is implemented, the multisig contract should be vetted to see if MI-01 Administrator Denial of Service and MI-02 Revoked-Administrator Signatures Are Still Valid need to be fixed in the multisig contract.
3. Make a rust macro with the multisig implementation and use it for all the administrator actions. If this recommendation is implemented, MI-01 Administrator Denial of Service and MI-02 Revoked-Administrator Signatures Are Still Valid need to be fixed in a single place, inside the macro.

Status

Not implemented. The development team informed us that they will not do this enhancement for this version.

EN-02 Administrator Signature Not Required on Execute

Location:

- `instructions/access_grant/access_grant_execute.rs: 9,22`
- `instructions/access_revoke/access_revoke_execute.rs: 9,22`
- `instructions/multisig_modify/multisig_modify_execute.rs: 9,22`
- `instructions/treasury_burn/treasury_burn_execute.rs: 15,28`
- `instructions/treasury_transfer/treasury_transfer_execute.rs: 12,25`
- `instructions/vesting_add/vesting_add_execute.rs: 11,24`
- `instructions/vesting_remove/vesting_remove_execute.rs: 11,24,31`
- `instructions/vesting_transfer/vesting_transfer_execute.rs: 12,25`

If the rent used to allocate the `multisig_instruction` is given back to the administrator that created the action request on execute (instead of the signer of the `*_execute` instruction), then it is possible to relax the `*_execute` instructions and let any account run them without any loss in security.

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 programs, owners or project investors.

Centralization

The analyzed program has 2 different functionalities, vesting and managing the operations vault from the operations wallet.

For the operations vault functionality, a group of operators, which include all the administrators, can send tokens to any address.

For the vesting functionality, a group of administrators has to accumulate enough signatures to handle the vesting schedules. This schema is also used for other administrative operations. See Administrators Actions for details.

It also must be noted that in Solana, depending on how the programs are deployed onto the network, the deployer may have rights to switch the program's code with a new one.

Privileged Roles

These are the privileged roles that we identified on the audited program.

It must be noted that any signer can claim the vesting tokens for any beneficiary via the `vesting_claim` instruction and the funds will be transferred to the beneficiary (not the signer of the instruction).

Operator

A signer with the operator role can transfer funds away in the `operations_vault` to any other account via the `operations_transfer` instruction.

Administrator

Besides all the actions that can be made by an operator, a signer with the administrator role can create, cancel, sign and execute instructions handled via the program's Administrators Actions.

Administrators Actions

Administrators actions are executed only if several administrators sign them. This is achieved by having 4 Solana instructions for each action. First an administrator will create an action request and sign it via the `*_create` instructions, then other administrators can sign the action request via the `*_sign` instructions. At any time any administrator can cancel the signing process via the `*_cancel` instructions. If enough administrators sign the action request, then an administrator can execute the action via the `*_execute` instructions. It must be noted that the program checks that enough administrators are available to sign action requests at all times.

The following actions are handled via this mechanism:

- Add new administrators or operators via the `access_grant_*` instructions.
- Remove administrators or operators via the `access_revoke_*` instructions.
- Set the number of administrators required to sign action requests via the `multisig_modify_*` instructions.
- Burn funds in the treasury vault via the `treasury_burn_*` instructions.
- Transfer funds owned by the treasury vault to any other account via the `treasury_transfer_*` instructions.
- Create a vesting schedule for a recipient via the `vesting_add_*` instructions.
- Remove a vesting schedule via the `vesting_remove_*` instructions.
- Transfer funds owned by the vesting valt to any other account via the `vesting_transfer_*` instructions.

Changelog

- 2022-09-09 – Initial report based on commit `f9bd6340f8e44dd8c799ef4b8e239b5aaea51cbb`.
- 2022-10-04 – Check fixes on commit `664661f94b6af229d7aeff9b997d5474d1920f51`.
- 2024-06-06 – Check changes on commit `c87bb7369ce63ad640587f72022a11d25aba968d`.

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