0xCommit

Security Audit Report

SOEX - Solana Programs

Staking LP , CVT and HVT Contracts

Version: Final

Date: 2 Nov 2024

Table of Contents

License

Disclaimer

THE CONTENT OF THIS AUDIT REPORT IS PROVIDED “AS IS”, WITHOUT REPRESENTATIONS AND WARRANTIES OF ANY KIND.

THE AUTHOR AND HIS EMPLOYER DISCLAIM ANY LIABILITY FOR DAMAGE ARISING OUT OF, OR IN CONNECTION WITH, THIS AUDIT REPORT.

Introduction

Purpose of this report

0xCommit has been engaged by **SOEX - Solana Programs** to perform a security audit of several Solana Programs components.

The objectives of the audit are as follows:

1. Determine the correct functioning of the protocol, in accordance with the project specification.
2. Determine possible vulnerabilities, which could be exploited by an attacker.
3. Determine solana program bugs, which might lead to unexpected behaviour.
4. Analyze whether best practices have been applied during development.
5. Make recommendations to improve code safety and readability.

This report represents a summary of the findings.

As with any code audit, there is a limit to which vulnerabilities can be found, and unexpected execution paths may still be possible. The author of this report does not guarantee complete coverage (see disclaimer).

Codebases Submitted for the Audit

The audit has been performed on the following GitHub repositories:

Version	List of programs	Source
1	Staking LP , CVT and HVT	https://gitlab.dcircle.io/dcchain/soex_solana_x/-/tree/master/programs

How to Read This Report

This report classifies the issues found into the following severity categories:

Severity	Description
Critical	A serious and exploitable vulnerability that can lead to loss of funds, unrecoverable locked funds, or catastrophic denial of service.
Major	A vulnerability or bug that can affect the correct functioning of the system, lead to incorrect states or denial of service.
Minor	A violation of common best practices or incorrect usage of primitives, which may not currently have a major impact on security, but may do so in the future or introduce inefficiencies.
Informational	Comments and recommendations of design decisions or potential optimizations, that are not relevant to security. Their application may improve aspects, such as user experience or readability, but is not strictly necessary. This category may also include opinionated recommendations that the project team might not share.

The status of an issue can be one of the following: **Pending**, **Acknowledged**, or **Resolved**.

Note that audits are an important step to improving the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of the system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**. We include a table with these criteria below.

Note that high complexity or low test coverage does not necessarily equate to a higher risk, although certain bugs are more easily detected in unit testing than in a security audit and vice versa.

Overview

Methodology

The audit has been performed in the following steps:

1. Gaining an understanding of the code base's intended purpose by reading the available documentation.
2. Automated source code and dependency analysis.
3. Manual line by line analysis of the source code for security vulnerabilities and use of best practice guidelines, including but not limited to:
    a. Race condition analysis
    b. Under-/overflow issues
    c. Key management vulnerabilities
4. Report preparation

Summary of Findings

S.N	Program	Description	Severity	Status
1	Staking LP	Signature hash considers inadequate parameters leading to abuse of signature.	High	Resolved
2	Staking LP	Freezing variable oversized.	Low	Acknowl...
3	Staking LP	No mechanism to change pool configuration.	Low	Resolved
4	HVT	Inited variable oversized.	Low	Acknowl...
5	HVT	No mechanism to change config	Low	Resolved
6	CVT	Init functions can be called multiple times	High	Resolved

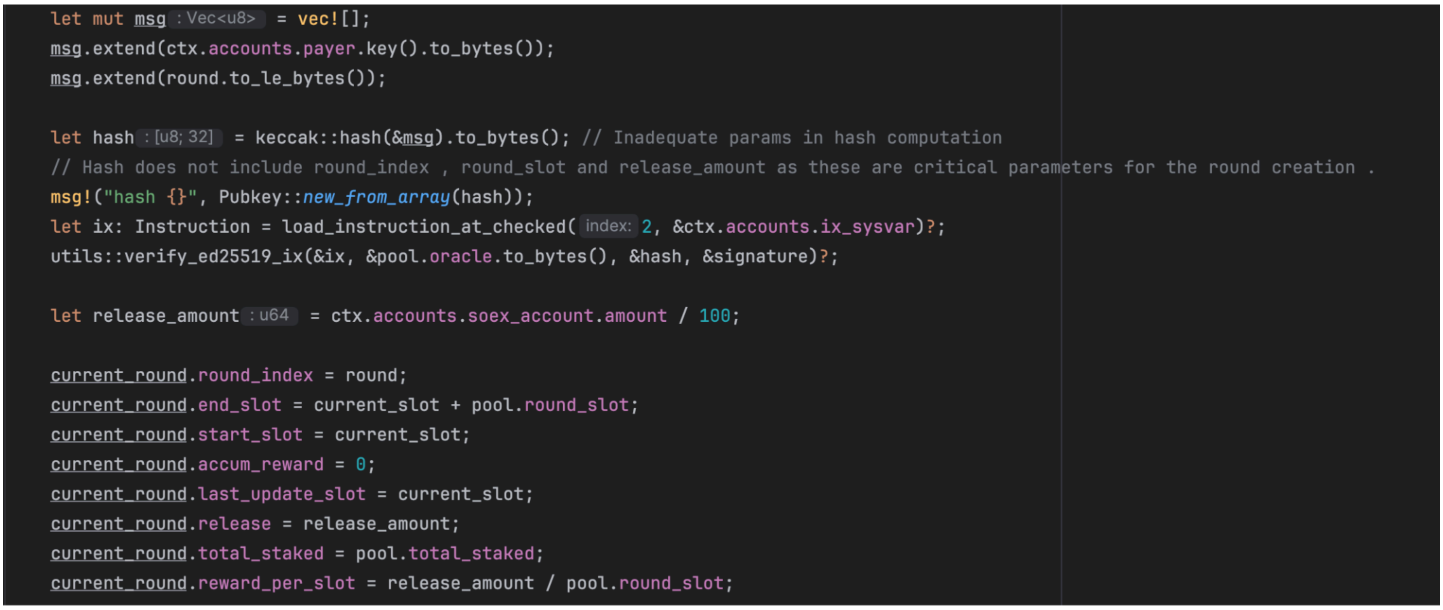
Detailed Findings

1. Signature hash considers inadequate parameters leading to abuse of signature.

Severity: High ▾ Program - Staking LP

Description

For `handle_oracle_round_create` a function hash is created with limited parameters namely `payers.key` and **round number**, Since limited parameters are used, same signature can be replayed to change/reset the other parameters.



```
let mut msg : Vec<u8> = vec![];
msg.extend(ctx.accounts.payer.key().to_bytes());
msg.extend(round.to_le_bytes());

let hash : [u8; 32] = keccak::hash(&msg).to_bytes(); // Inadequate params in hash computation
// Hash does not include round_index , round_slot and release_amount as these are critical parameters for the round creation .
msg!("hash {}", Pubkey::new_from_array(hash));
let ix: Instruction = load_instruction_at_checked( index: 2, &ctx.accounts.ix_sysvar)?;
utils::verify_ed25519_ix(&ix, &pool.oracle.to_bytes(), &hash, &signature)?;

let release_amount : u64 = ctx.accounts.soex_account.amount / 100;

current_round.round_index = round;
current_round.end_slot = current_slot + pool.round_slot;
current_round.start_slot = current_slot;
current_round.accum_reward = 0;
current_round.last_update_slot = current_slot;
current_round.release = release_amount;
current_round.total_staked = pool.total_staked;
current_round.reward_per_slot = release_amount / pool.round_slot;
```

Remediation

Add `round_index`, `round_slot` and `release_amount` while computing hash such that signature reuse does not happen.

Status

Resolved ▾

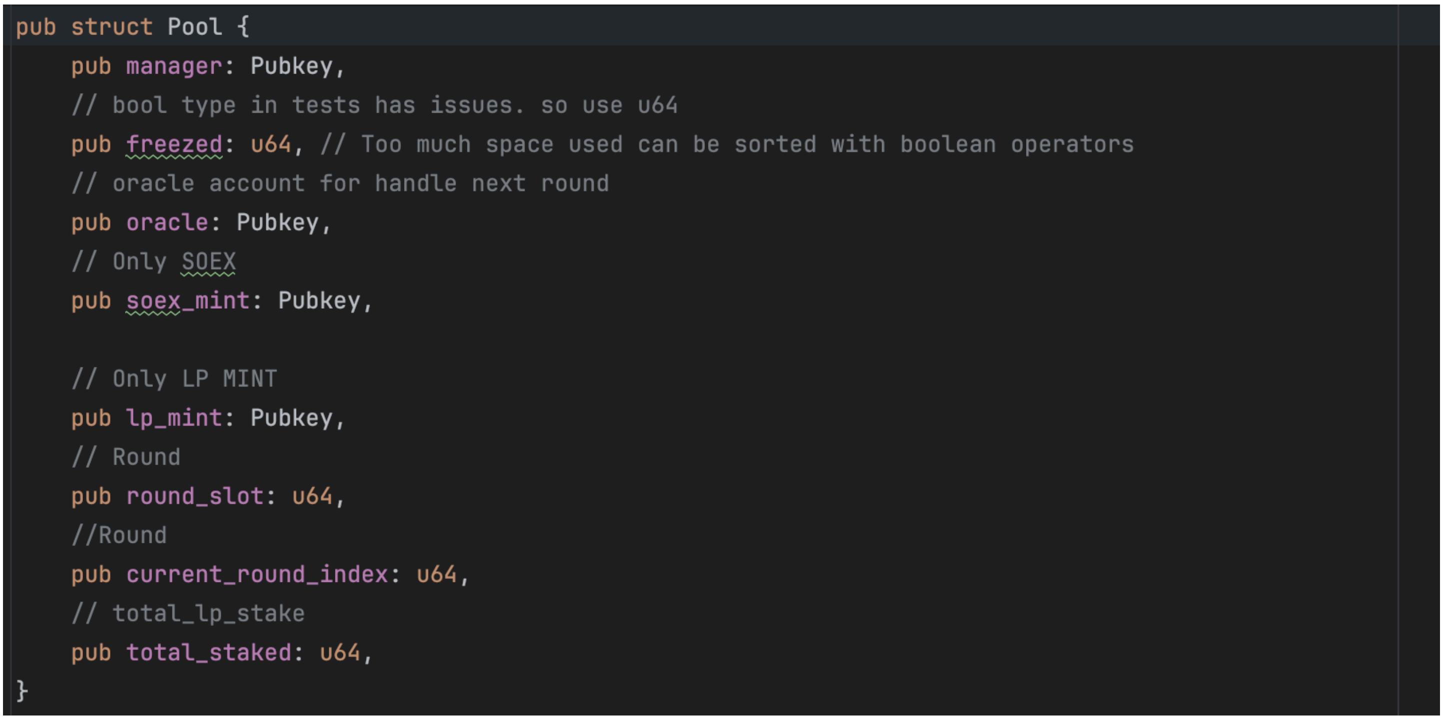
2. Freezing variable oversized.

Severity: Low

Program - Staking LP

Description

The state variable handling freezing uses `u64` which is too big for the task it performs.



```
pub struct Pool {
    pub manager: Pubkey,
    // bool type in tests has issues. so use u64
    pub freezed: u64, // Too much space used can be sorted with boolean operators
    // oracle account for handle next round
    pub oracle: Pubkey,
    // Only SOEX
    pub soex_mint: Pubkey,

    // Only LP MINT
    pub lp_mint: Pubkey,
    // Round
    pub round_slot: u64,
    //Round
    pub current_round_index: u64,
    // total_lp_stake
    pub total_staked: u64,
}
```

Remediation

Freeze variable can be reduced down to `bool` from `u64`.

Status

Acknowledged

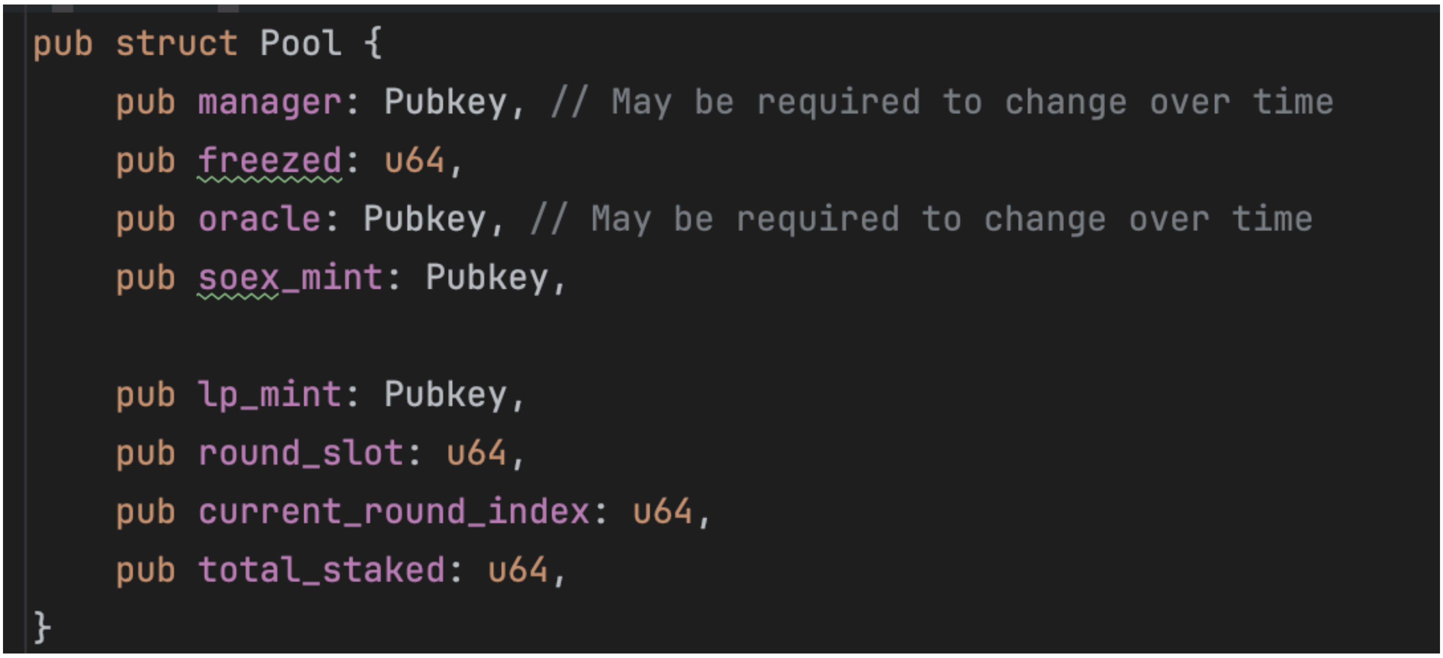
3. No mechanism to change pool configuration

Severity: Low   Program - Staking LP

Description

The config parameters of `Pool` once set during initialization can’t be edited. This functionality is needed for stability of the Staking LP program. As parameters can change over the life cycle of the program.

Following parameters in `Pool` may require alteration over time.



```
pub struct Pool {
    pub manager: Pubkey, // May be required to change over time
    pub freezed: u64,
    pub oracle: Pubkey, // May be required to change over time
    pub soex_mint: Pubkey,

    pub lp_mint: Pubkey,
    pub round_slot: u64,
    pub current_round_index: u64,
    pub total_staked: u64,
}
```

Remediation

Add functions which allow modification of `Pool`. (Note - These changes are subject to admin control)

Status

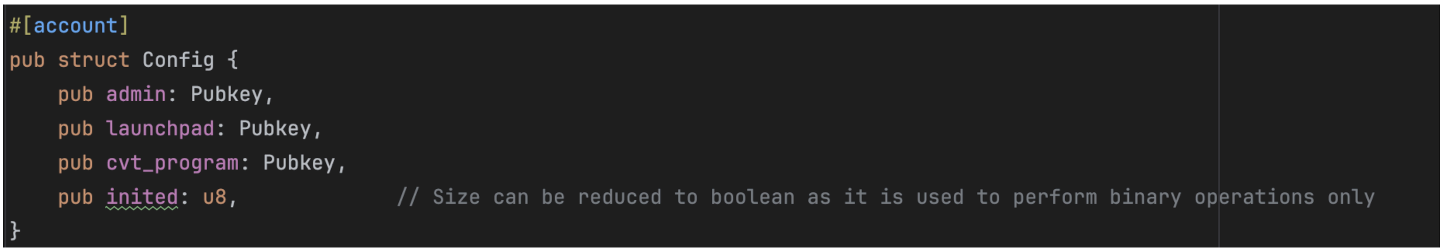
Resolved

4. `Inited` variable oversized.

Severity: Low ▾ Program - HVT

Description

The state variable `inited` uses `u8` which is too big for the task it performs. It can be reduced down to boolean.



```
#[account]
pub struct Config {
    pub admin: Pubkey,
    pub launchpad: Pubkey,
    pub cvt_program: Pubkey,
    pub inited: u8,            // Size can be reduced to boolean as it is used to perform binary operations only
}
```

Remediation

Freeze variable can be reduced down to `bool` from `u64`.

Status

Acknowledged ▾

5. `Init` Function can be called multiple times

Severity: **High** ▾ **Program - CVT**

Description

The `init_soex_acct` function is part of the initialization process and ideally it needs to be called once, Since it's not part of the initialize function presented by default it can be called over multiple times and there is no mechanism to prevent it. It must be ensured that all functions associated with the initialization process must be called only once.

Remediation

Add a state (i.e. - `InitCompleted [boolean]`) which gets disabled once `init_soex_acct` functions are called and which makes `init_soex_acct` function use once only.

Status

Resolved ▾

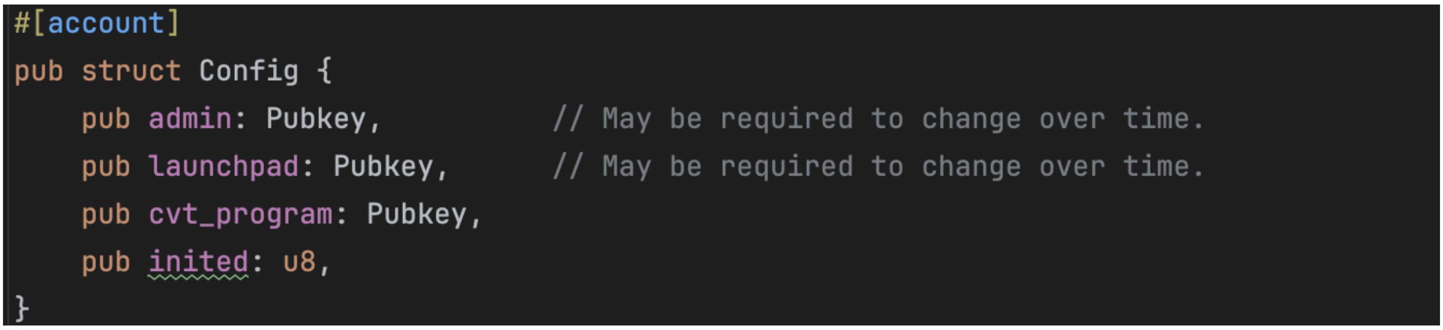
6. No mechanism to change Config

Severity: Medium ▾ Program - HVT

Description

The `Config` parameters set during initialization can't be edited. This functionality is needed for stability of the CVT program. As parameters can change over the life cycle of the program.

Following parameters in config may require alteration over time.



```
#[account]
pub struct Config {
    pub admin: Pubkey,           // May be required to change over time.
    pub launchpad: Pubkey,       // May be required to change over time.
    pub cvt_program: Pubkey,
    pub inited: u8,
}
```

Remediation

Add functions which allow modification of `Config`. (Note - These changes are subject to admin control)

Status

Resolved ▾