TORII

RTR Staking and Lottery Solana Program Code Review and Security Analysis

Client: Restore the Republic

Date: 2nd September 2024

Version: 1.0

Table of Contents

Security Review Information

Repository	rtr-staking
Initial commit	dad4570e5320eff5615a13ef8d9424fbb5c04569
Final commit	5576e50a87b2f83bf694bc6decd9712b7c6acdf7
Scope	programs/rtr_staking/* programs/rtr_lottery/*
Version	Final Report (v. 1.0)
Date	2nd September. 2024

Executive summary

As of 2nd September. 2024, our comprehensive security review of the RTR Token Staking and Lottery programs has been concluded. The assessment identified 1 medium-severity, 1 low-severity, and 2 informational issues. We are pleased to report that, following our review and the implementation of recommended fixes, all identified issues have been successfully resolved.

Introduction

Torii Security has been commissioned by Restore the Republic to conduct a comprehensive security review of their Solana Program, focusing on the robustness, security, and efficiency of the program's implementation. The review aims to critically evaluate the program's architecture and codebase, with the following specific objectives:

- **Verify Protocol Integrity**: Assess the program's operation against its design specifications to ensure it functions correctly and efficiently within the Solana ecosystem. This includes evaluating its interaction with other protocols and services on the Solana network.
- **Identify Security Vulnerabilities**: Uncover potential security weaknesses that could be exploited by malicious actors, including but not limited to, flaws in program logic, transaction handling, and external dependencies.
- **Detect Program Bugs**: Identify bugs and glitches in the code that may result in unintended or erratic program behavior, potentially compromising its performance or security.
- **Provide Improvement Recommendations**: Offer actionable advice to enhance the program's security posture, efficiency, and code clarity, aiming to fortify it against current and future security threats while improving maintainability and scalability.

Exclusions from This Review

While the security review conducted by Torii Security on behalf of Restore the Republic provides a comprehensive analysis of the Solana Program's architecture, codebase, and

security posture, certain aspects are beyond the scope of this audit. These exclusions are critical for stakeholders to understand, as they may require separate consideration and evaluation. The following areas have not been verified as part of this security review:

- **Deployment and Program Upgrade Process:** The procedures and mechanisms for deploying the Solana Program to the network and subsequent upgrades or modifications to the program are not covered. This includes the validation of deployment scripts, migration strategies, and the security of upgradeable contract mechanisms.
- **Keys Management**: The management, storage, and security practices for keys, including administrator keys and those used for program interactions, are outside the scope of this review. This encompasses both the technical and procedural safeguards in place to protect keys from unauthorized access or misuse.
- **Economic Vulnerabilities**: The review does not delve into the economic aspects or incentive structures of the RTR protocol. Potential vulnerabilities arising from economic models, tokenomics, or financial incentives that could impact the program's security or integrity are not evaluated.

Protocol Overview

The RTR Token Staking and Lottery System is a decentralized program on the Solana blockchain designed to incentivize token holders by allowing them to stake RTR tokens in exchange for rewards. The system also features a lottery mechanism integrated with the staking program, offering users a chance to win additional prizes based on their staking positions. The protocol is built with security and fairness in mind, utilizing verifiable random functions (VRF) for lottery operations and ensuring transparent and secure reward distribution.

Key Features

Staking Program

- **Staking Pool Initialization:** The staking pool is initialized by an admin, setting up global parameters, including the total rewards and reward rate.
- **RTR Token Staking:**Users can stake RTR tokens, which are then transferred to a program-controlled account, and their staking positions are updated.
- **RTR Token and Reward Withdrawal:** Users can withdraw their staked tokens along with the accumulated rewards at any time. The rewards are calculated and transferred accordingly.
- **Reward Claiming:** Users have the option to claim their accumulated rewards without needing to unstake their tokens.
- **Reward Calculation:** The system continuously accumulates rewards over a fixed 270-day period, with the reward rate adjusted dynamically based on the total staked amount.

Lottery

- **Lottery Settings Initialization:** Admin initializes the lottery settings, including setting up the prize token mint and configuring other global parameters.
- **Start New Lottery:** A new lottery round is started by the authority, where the prize amount is transferred to an escrow account.
- **Roll Lottery (Commit to Randomness):** The lottery authority commits to randomness by initiating a roll, which involves capturing an IPFS snapshot of all stakers and recording a commit slot.
- **Pick Winner:** Using Switchboard's verifiable random function (VRF), the system generates a random number to pick a lottery winner. The process is transparent and verifiable.
- **Release Lottery Funds:** Once a winner is selected, the authority releases the funds to the winner, and the lottery is marked as ended.
- **Modify Lottery Authority:** The authority of the lottery system can be transferred to a new entity if needed, ensuring flexible control over the system.

Security Considerations

- **Access Control for Admin Functions:** Critical functions such as initialization and lottery control are restricted to authorized administrators only.

- **Secure Token Transfers:** All token transfers are securely managed through Cross-Program Invocations (CPIs) to ensure the safety of users' assets.
- **Verifiable Randomness and Fairness:** The use of Switchboard VRF and immutable IPFS snapshots ensures that the lottery process is both fair and transparent, with results that can be independently verified by any participant.

Risks

The RTR Token Staking and Lottery System involves several potential risks that need to be carefully managed to ensure the security and reliability of the protocol. Below is a detailed assessment of these risks, categorized into technical, operational, and economic risks.

Technical Risks

- **Overflow and Underflow in Calculations**
  - **Risk:** Arithmetic overflows or underflows during reward calculations or token transfers could lead to incorrect distribution of rewards or unintended behaviors.
  - **Mitigation:** Use of safe arithmetic libraries and thorough testing of edge cases to prevent such errors.
- **Randomness Manipulation**
  - **Risk:** The randomness used in the lottery system could be manipulated if the VRF is compromised or the process is not properly secured.
  - **Mitigation:** Utilize a trusted verifiable random function (VRF) like Switchboard and ensure the process is transparent and verifiable by participants.
- **Security of Off-Chain Components**
  - **Risk:** The off-chain script used for winner selection could be tampered with, leading to unfair outcomes.
  - **Mitigation:** Open-source the script to allow public scrutiny and verification, and ensure that all critical components are independently auditable.

Operational Risks

- **Access Control Breaches**
  - **Risk:** Unauthorized access to admin functions could lead to malicious activities, such as unauthorized withdrawal of funds or manipulation of the staking pool.
  - **Mitigation:** Implement strong access controls, use multi-signature wallets for critical operations, and regularly review and update access permissions.
- **Dependency on External Oracles**
  - **Risk:** The reliance on external oracles like Switchboard for randomness introduces dependency risks, where failure or compromise of the oracle could affect the protocol.
  - **Mitigation:** Diversify reliance on oracles by incorporating fallback mechanisms and regularly monitoring the performance and security of the chosen oracle service.
- **Unintended Bugs in Off-Chain Components**

  - **Risk:** Bugs in the off-chain script for winner selection or IPFS snapshot management could lead to incorrect or unfair outcomes.
  - **Mitigation:** Conduct rigorous testing, peer reviews, and maintain transparency by open-sourcing the code to enable community oversight.

Economic Risks

- **Reward Rate Misconfiguration**
  - **Risk:** Incorrect configuration of the reward rate could lead to unsustainable reward payouts, depleting the reward pool prematurely or diluting rewards for users.
  - **Mitigation:** Carefully calculate and periodically review the reward rate settings, taking into account the total staked amount and expected user participation.
- **Liquidity Risks**
  - **Risk:** Insufficient liquidity in the staking pool or prize fund could prevent users from withdrawing their staked tokens or rewards.
  - **Mitigation:** Maintain adequate reserves, monitor liquidity levels, and implement safeguards to ensure the availability of funds for withdrawals.
- **Economic Exploits**
  - **Risk:** Economic exploits such as Sybil attacks or staking manipulation (e.g., flash staking) could be used to game the reward system or lottery.
  - **Mitigation:** Implement anti-Sybil measures, enforce minimum staking periods, and consider mechanisms to detect and prevent exploitative behavior.

Fairness and Transparency Risks

- **IPFS Snapshot Integrity**
  - **Risk:** If the IPFS snapshot of stakers is compromised or manipulated, it could lead to unfair lottery outcomes.
  - **Mitigation:** Ensure the snapshot process is transparent, securely timestamped, and verifiable by all participants.
- **Centralization of Authority**
  - **Risk:** Concentration of control in a single authority or admin could lead to abuse of power or manipulation of the protocol.
  - **Mitigation:** Decentralize control where possible, use multi-signature governance for critical actions, and engage the community in decision-making processes.

Methodology

Issue Severity Classification

This report differentiates identified issues into distinct severity levels, each reflecting the potential impact on the system's security and overall functionality.

Severity	Description
Critical	Issues that present an immediate and severe threat, such as significant financial loss, irreversible locking of funds, or catastrophic system failure. These vulnerabilities require urgent remediation.
High	Bugs or vulnerabilities that could disrupt the correct operation of the system, potentially leading to incorrect states or temporary denial of service. Prompt attention and corrective action are necessary.
Medium	Issues that indicate deviations from best practices or suboptimal use of system primitives. While they may not pose immediate security threats, these issues could lead to vulnerabilities or inefficiencies if unaddressed.
Low	Minor concerns that have a negligible impact on system security or functionality. These may include inefficiencies or minor deviations from best practices that are unlikely to affect the system's operation significantly.
Informational	Suggestions related to design decisions, potential enhancements, or optimizations that do not have a direct impact on security. Implementing these recommendations may improve aspects such as usability or code readability but is not essential for system security.

Issue Status Definitions

Each issue is assigned a status reflecting its current resolution stage.

Status	Description
Pending	The issue has been identified but not yet reviewed or addressed by the development team.
Acknowledged	The development team has recognized the issue but has not completed its resolution.
Resolved	The issue has been fully addressed, with implemented changes verified for effectiveness.

Findings Summary

ID	Title	Severity	Status
TS-H1	Incomplete Validation in pick_winner Due to Missing Randomness Account Reference	Medium	**Fixed**
TS-L1	Lack of On-Chain Verification for Lottery Prize Distribution	Low	**Mitigated**
TS-I2	Code Quality and Optimization Improvements	Informational	**Fixed**
TS-I2	Potential Initialization Frontrunning	Informational	**Fixed**

Detailed Findings

TS-M1 - Incomplete Validation in pick_winner Due to Missing Randomness Account Reference

Classification

Severity: Medium
Status: Fixed

Description

In the current implementation of the roll instruction within the RTR Token Lottery System, the account responsible for providing randomness is not stored in the **Lottery** account. As a result, during the ***pick_winner*** instruction, the system only validates the commit slot, without verifying the specific randomness account used.

This creates a potential vulnerability where the authority could manipulate the outcome by creating multiple randomness accounts and selecting the one that produces a favorable result after observing the commit slot's effect. This undermines the fairness of the lottery by allowing the authority to potentially pick a winner unfairly.

Recommendation

To ensure the integrity and fairness of the lottery process, the randomness account used during the roll instruction should be saved in the Lottery account. This way, during the ***pick_winner*** instruction, the system can verify that the same randomness account is used, preventing any possibility of manipulation.

Remediation

The issue was fixed in the 5576e5 commit by verifying the randomness data account in the ***pick_winner*** instruction.

TS-L1 - Lack of On-Chain Verification for Lottery Prize Distribution

Classification

Severity: Low
Status: **Mitigated**

Description

The RTR Token Lottery System adopts an optimistic architecture where the winner of the lottery is determined off-chain using the ***RandomnessAccountData*** provided by Switchboard's on-demand service. While this approach leverages external randomness to ensure fairness, the actual selection and distribution of the prize rely on off-chain processes controlled by the protocol's administrator.

After the off-chain determination of the winner, the protocol's administrator is responsible for executing lottery-related instructions, including the release of the prize to the designated winner. However, there is no on-chain verification to ensure that the prize is transferred to the legitimately selected winner. Although the protocol is designed to be transparent and verifiable, allowing anyone to audit the process and confirm that the correct winner was selected, this mechanism relies heavily on off-chain trust.

Recommendation

Modify the lottery architecture to include on-chain verification of the selected winner before any prize transfer is executed. This could involve storing the winning address on-chain after the off-chain selection and requiring that the prize transfer be validated against this stored address.

Remediation

The issue was mitigated with the RTR team comment:

The current lottery system relies on a complex, computation-intensive off-chain process for winner selection, which is not feasible to execute on-chain due to resource limitations. To ensure security and fairness, the system operates under the following trust assumptions: ***the lottery authority is a trusted entity governed by multisig to prevent unilateral control****, and* ***the prize distribution process is fully auditable****, with the winner selected based on immutable data and a transparent, verifiable algorithm. These measures provide a balanced approach, ensuring the lottery's integrity despite the reliance on off-chain components.*

TS-I1 - Code Quality and Optimization Improvements

Classification

Severity: Informational
Status: **Fixed**

Description

The following areas within the RTR Token Staking Program's codebase present opportunities for optimization and improvements in code quality:

1. **Unnecessary Result Type in should_close Function:**
   - The should_close function, found in *programs/rtr-staking/src/state/user_staking_position.rs*, returns a Result<bool>. However, the function's implementation does not actually return an Err variant, making the Result type unnecessary. This function can be simplified to return a plain bool instead.
2. **Redundant Mutability and Useless Print Statements in update_rewards Function:**
   - The update_rewards function in programs/rtr-staking/src/state/user_staking_position.rs takes a mutable reference to the StakingPool type, though this reference is never actually modified within the function, rendering the mutability redundant. The function should instead take a regular reference to StakingPool. Additionally, the function includes println! macro calls, which are irrelevant in the Solana execution environment and should be removed.

Recommendation

Simplify should_close Return Type:

- Change the return type of the should_close function from Result<bool> to bool to remove unnecessary complexity.

Refactor update_rewards Function:

- Modify the update_rewards function to accept a regular reference to StakingPool instead of a mutable one. Additionally, remove all println! macro calls to avoid unnecessary operations in the Solana execution environment.

Remediation

The issue was fixed in the 5576e5 commit by implementing recommended modifications.

TS-I2 - Potential Initialization Frontrunning

Classification

Severity: Informational
Status: **Fixed**

Description

The StakingPool account, which stores all critical information related to the staking protocol, is a Program Derived Address (PDA) generated using a single hard-coded seed for address derivation. This design introduces a vulnerability where the first entity to call the initialize instruction gains full control over the protocol's configuration. If a malicious actor manages to frontrun this initialization, they would have the ability to configure the protocol maliciously, requiring a full redeployment of the program to recover from the attack.

Recommendation

It is recommended to either verify authority that can initialize staking pool, or introduce the process to verify if protocol was initialized by correct authority.

Remediation

The issue was fixed. RTR team comment:

An off-chain procedure is in place to validate whether the StakingPool was initialized by an unauthorized entity. If such an unauthorized initialization is detected, the program will be closed, and the funds required for deployment will be returned to the protocol authority. This ensures that any malicious attempts to initialize the protocol can be swiftly mitigated, preserving the integrity of the system.

Disclaimer

This report has been prepared in accordance with the current best practices and standards applicable at the time of its preparation. It is intended to provide an analysis and evaluation of the subject matter based on the information available, including any potential vulnerabilities, issues, or risks identified during the assessment process. The scope of this analysis is limited to the data, documents, and materials provided for review, and the conclusions drawn are based on the status of the information at the time of the report.

No representation or warranty, express or implied, is made as to the absolute completeness or accuracy of the information contained in this report. It is important to acknowledge that the findings, conclusions, and recommendations presented are subject to change should there be any modifications to the subject matter. This report should not be viewed as a conclusive or exhaustive evaluation of the subject matter's safety, functionality, or reliability. Stakeholders are encouraged to conduct their own independent reviews, assessments, and validations to ensure the integrity and security of the evaluated subject.

The authors and auditors of this report disclaim any liability for any direct, indirect, incidental, or consequential damages or losses that may result from reliance on this report or its contents. It is the responsibility of the stakeholders to ensure the ongoing monitoring and evaluation of the subject matter to mitigate potential risks or vulnerabilities.

The nature of technology and digital systems means that they are inherently subject to risks, including but not limited to vulnerabilities, bugs, and security threats that may not be foreseeable at the time of this report. The dynamic and evolving nature of technological standards, security practices, and threat landscapes means that absolute security cannot be guaranteed. Despite thorough analysis and evaluation, unforeseen vulnerabilities may exist, and new threats may emerge subsequent to the issuance of this report.

The authors and auditors make no guarantee regarding the impenetrability or infallibility of the subject matter under evaluation. Stakeholders are advised to implement comprehensive security measures, including but not limited to regular updates, patches, and monitoring, to safeguard against potential threats and vulnerabilities.