BEOSIN
Web3 Security & Compliance

Sonic-Staking

Smart Contract Security Audit

No. 202501061600

Jan 6th, 2025

SECURING BLOCKCHAIN ECOSYSTEM

WWW.BEOSIN.COM

Contents

Summary of Audit Results

After auditing, 1 Low risk, 1 Info items were identified in the Sonic-Staking project. Specific audit details will be presented in the Findings section. Users should pay attention to the following aspects when interacting with this project:

Low	Fixed : 1	Acknowledged: 0
Info	Fixed : 0	Acknowledged: 1

- **Project Description:**

The Sonic-Staking project consists of three core functionalities: reward addition, airdrop staking, and wallet staking. Below is a more detailed description:

The Withdraw Pool Supply and Add Pool Supply contracts implement the functions of extracting and supplying funds to the staking pool. The Withdraw Pool Supply contract allows the designated withdrawal_authority address to withdraw reward tokens from the staking pool's staking_pool_token_account. Before the withdrawal, the contract performs permission verification, pool status checks, and validity checks on the withdrawal amount to ensure compliance with the rules. After a successful withdrawal, the contract updates the staking pool's state and records the details of the operation through events, including the withdrawal amount and account information. The Add Pool Supply contract allows the supply_provider to supply additional tokens to the staking pool account. The contract verifies that the staking pool is enabled and ensures the supply amount is valid. After a successful transfer, the contract updates the staking pool's state and records detailed information about the supply, such as the supplied amount and the provider's information, ensuring the transparency and traceability of fund management.

The Airdrop Staking and Airdrop Staking Withdraw functions are implemented by the airdrop_staking.rs and airdrop_staking_withdraw.rs contracts, which allow the project team to stake tokens on behalf of users through the back_authority account and support withdrawal operations. The Airdrop Staking contract allows the back_authority account to stake a certain amount of tokens into a specified staking pool on behalf of the user, with no need for the user to provide tokens. The contract sets parameters such as staking amount, reward amount, staking period, etc., and performs a series of checks to ensure that the staking operation complies with the requirements, such as verifying if the staking pool is enabled and if the reward pool has sufficient tokens. After successful staking, the contract updates the staking pool and user information and triggers the staking event. The Airdrop Staking Withdraw contract allows users to withdraw staked tokens and rewards. Before withdrawal, the contract verifies that the staking period has ended, ensures the user has applied for withdrawal, and confirms that the corresponding fees have been paid to the fee_receiver address. After a successful withdrawal, the contract transfers the tokens to the user's account, deducts the fee, updates the staking pool and user status, and triggers the withdrawal event.

The Wallet Staking and Wallet Staking Withdraw functions are implemented by the wallet_staking.rs and wallet_staking_withdraw.rs contracts, allowing users to stake tokens into a specified staking pool

and withdraw staked tokens and rewards after the staking period ends. The `Wallet Staking` contract allows users to transfer tokens into the contract and specify staking amount, reward amount, staking period, and other parameters. The contract performs a series of checks, including verifying if the staking pool is enabled, confirming that the staking amount meets the minimum requirement, and ensuring that the reward amount is reasonable and in accordance with the pool's distribution rules. After successful staking, the contract updates the staking pool and user information and triggers the staking event. The `Wallet Staking Withdraw` contract allows users to withdraw their staked tokens and rewards after the staking period has ended. Before withdrawal, the contract verifies that the staking period has expired, checks whether the user has already applied for withdrawal, and ensures the withdrawal amount is reasonable and that the staking pool has sufficient balance to cover the withdrawal. After a successful withdrawal, the contract updates the staking pool and user information, triggers the withdrawal event, and records the details of the operation.

1 Overview

1.1 Project Overview

Project Name	Sonic-Staking
Project Language	Rust
Platform	Solana
Github Link	https://github.com/mirrorworld-universe/sonic-staking-program-library
Commit	93409c4227b48d583937300132c020927110a61a(initial) 456cead5acf63e354dc81fb773f23db5a9ae1b80 717ecbc83452789ffcea9c99ef3f94b911cd6611(final)

1.2 Audit Overview

Audit work duration: Dec 31, 2024 – Jan 6, 2025

Audit team: Beosin Security Team

1.3 Audit Method

The audit methods are as follows:

1. Formal Verification

Formal verification is a technique that uses property-based approaches for testing and verification. Property specifications define a set of rules using Beosin's library of security expert rules. These rules call into the contracts under analysis and make various assertions about their behavior. The rules of the specification play a crucial role in the analysis. If the rule is violated, a concrete test case is provided to demonstrate the violation.

2. Manual Review

Using manual auditing methods, the code is read line by line to identify potential security issues. This ensures that the contract's execution logic aligns with the client's specifications and intentions, thereby safeguarding the accuracy of the contract's business logic.

The manual audit is divided into three groups to cover the entire auditing process:

The Basic Testing Group is primarily responsible for interpreting the project's code and conducting comprehensive functional testing.

The Simulated Attack Group is responsible for analyzing the audited project based on the collected historical audit vulnerability database and security incident attack models. They identify potential attack vectors and collaborate with the Basic Testing Group to conduct simulated attack tests.

The Expert Analysis Group is responsible for analyzing the overall project design, interactions with third parties, and security risks in the on-chain operational environment. They also conduct a review of the entire audit findings.

3. Static Analysis

Static analysis is a method of examining code during compilation or static analysis to detect issues. Beosin-VaaS can detect more than 100 common smart contract vulnerabilities through static analysis, such as reentrancy and block parameter dependency. It allows early and efficient discovery of problems to improve code quality and security.

2 Findings

Index	Risk description	Severity level	Status
Sonic-Staking-01	Centralization Risk in Withdraw Function	Low	Fixed
Sonic-Staking-02	Missing Reward Transfer in Pool Initialization	Info	Acknowledged

Finding Details:

[Sonic-Staking-01] Centralization Risk in Withdraw Function

Severity Level	Low
Lines	src\instructions\withdraw_pool_supply.rs#L70-140
Type	Business Security
Description	In the `handle_withdraw_pool_supply` function, the current implementation allows the `withdrawal_authority` address to withdraw the entire balance of the `staking_pool_token_account`, which poses a centralization risk. Since the `staking_pool_token_account` stores both reward tokens and users' staked tokens, the `withdrawal_authority` address should not have the ability to withdraw the entire balance.
Recommendation	It is recommended to restrict this permission, allowing the `withdrawal_authority` to only withdraw the excess reward portion.
Status	**Fixed.** The project owner has modified the withdrawal logic, and currently only the excess rewards can be withdrawn.

```
pub fn handle_withdraw_pool_supply(
    ctx: Context<WithdrawPoolSupplyInputAccounts>,
    params: &WithdrawPoolSupplyInputParams,
) -> Result<()> {
    let timestamp = Clock::get().unwrap().unix_timestamp;
    let staking_pool: &Box<Account<StakingPoolAccount>> =
&ctx.accounts.staking_pool;
    // Check
    check_withdrawal_authority(
        staking_pool.withdrawal_authority,
        ctx.accounts.withdrawal_authority.key(),
    )?;
    check_is_staking_pool_enable(staking_pool.enable)?;
    check_value_is_zero(params.amount as usize)?;
    check_pool_supply(ctx.accounts.staking_pool_token_account.amount
, params.amount)?;
```

[Sonic-Staking-02] Missing Reward Transfer in Pool Initialization

Severity Level	Info
Lines	src\instructions\initialize_staking_pool.rs
Type	Business Security
Description	In the `handle_initialize_staking_pool` function, the reward tokens were not transferred to the `staking_pool_token_account` when initializing the `staking_pool`. This may cause the user to receive rewards from other users' staking rewards when staking.
Recommendation	It is recommended to transfer the corresponding amount of `total_allocated_reward_amount` tokens to the account during the initialization of the staking_pool.
Status	**Acknowledged.**

3 Appendix

3.1 Vulnerability Assessment Metrics and Status in Smart Contracts

3.1.1 Metrics

In order to objectively assess the severity level of vulnerabilities in blockchain systems, this report provides detailed assessment metrics for security vulnerabilities in smart contracts with reference to CVSS 3.1(Common Vulnerability Scoring System Ver 3.1).

According to the severity level of vulnerability, the vulnerabilities are classified into four levels: "critical", "high", "medium" and "low". It mainly relies on the degree of impact and likelihood of exploitation of the vulnerability, supplemented by other comprehensive factors to determine of the severity level.

Likelihood \ Impact	Severe	High	Medium	Low
Probable	Critical	High	Medium	Low
Possible	High	Medium	Medium	Low
Unlikely	Medium	Medium	Low	Info
Rare	Low	Low	Info	Info

3.1.2 Degree of impact

- **Critical**

Critical impact generally refers to the vulnerability can have a serious impact on the confidentiality, integrity, availability of smart contracts or their economic model, which can cause substantial economic losses to the contract business system, large-scale data disruption, loss of authority management, failure of key functions, loss of credibility, or indirectly affect the operation of other smart contracts associated with it and cause substantial losses, as well as other severe and mostly irreversible harm.

- **High**

High impact generally refers to the vulnerability can have a relatively serious impact on the confidentiality, integrity, availability of the smart contract or its economic model, which can cause a greater economic loss, local functional unavailability, loss of credibility and other impact to the contract business system.

- **Medium**

Medium impact generally refers to the vulnerability can have a relatively minor impact on the confidentiality, integrity, availability of the smart contract or its economic model, which can cause a small amount of economic loss to the contract business system, individual business unavailability and other impact.

- **Low**

Low impact generally refers to the vulnerability can have a minor impact on the smart contract, which can pose certain security threat to the contract business system and needs to be improved.

3.1.3 Likelihood of Exploitation

- **Probable**

Probable likelihood generally means that the cost required to exploit the vulnerability is low, with no special exploitation threshold, and the vulnerability can be triggered consistently.

- **Possible**

Possible likelihood generally means that exploiting such vulnerability requires a certain cost, or there are certain conditions for exploitation, and the vulnerability is not easily and consistently triggered.

- **Unlikely**

Unlikely likelihood generally means that the vulnerability requires a high cost, or the exploitation conditions are very demanding and the vulnerability is highly difficult to trigger.

- **Rare**

Rare likelihood generally means that the vulnerability requires an extremely high cost or the conditions for exploitation are extremely difficult to achieve.

3.1.4 Fix Results Status

Status	Description
Fixed	The project party fully fixes a vulnerability.
Partially Fixed	The project party did not fully fix the issue, but only mitigated the issue.
Acknowledged	The project party confirms and chooses to ignore the issue.

3.2 Audit Categories

No.	Categories	Subitems
1	Coding Conventions	SPL Token Standards
		Visibility Specifiers
		Lamport Check
		Account Check
		Signer Check
		Program Id Check
		Deprecated Items
		Redundant Code
2	General Vulnerability	Integer Overflow/Underflow
		Reentrancy
		Pseudo-random Number Generator (PRNG)
		Transaction-Ordering Dependence
		DoS (Denial of Service)
		Function Call Permissions
		Returned Value Security
		Replay Attack
		Overriding Variables
		Third-party Protocol Interface Consistency
3	Business Security	Business Logics
		Business Implementations
		Manipulable Token Price
		Centralized Asset Control
		Asset Tradability
		Arbitrage Attack

Beosin classified the security issues of smart contracts into three categories: Coding Conventions, General Vulnerability, Business Security. Their specific definitions are as follows:

- **Coding Conventions**

Audit whether smart contracts follow recommended language security coding practices. For example, smart contracts developed in Solidity language should fix the compiler version and do not use deprecated keywords.

- **General Vulnerability**

General Vulnerability include some common vulnerabilities that may appear in smart contract projects. These vulnerabilities are mainly related to the characteristics of the smart contract itself, such as integer overflow/underflow and denial of service attacks.

- **Business Security**

Business security is mainly related to some issues related to the business realized by each project, and has a relatively strong pertinence. For example, whether the lock-up plan in the code match the white paper, or the flash loan attack caused by the incorrect setting of the price acquisition oracle.

*Note that the project may suffer stake losses due to the integrated third-party protocol. This is not something Beosin can control. Business security requires the participation of the project party. The project party and users need to stay vigilant at all times.

3.3 Disclaimer

The Audit Report issued by Beosin is related to the services agreed in the relevant service agreement. The Project Party or the Served Party (hereinafter referred to as the "Served Party") can only be used within the conditions and scope agreed in the service agreement. Other third parties shall not transmit, disclose, quote, rely on or tamper with the Audit Report issued for any purpose.

The Audit Report issued by Beosin is made solely for the code, and any description, expression or wording contained therein shall not be interpreted as affirmation or confirmation of the project, nor shall any warranty or guarantee be given as to the absolute flawlessness of the code analyzed, the code team, the business model or legal compliance.

The Audit Report issued by Beosin is only based on the code provided by the Served Party and the technology currently available to Beosin. However, due to the technical limitations of any organization, and in the event that the code provided by the Served Party is missing information, tampered with, deleted, hidden or subsequently altered, the audit report may still fail to fully enumerate all the risks.

The Audit Report issued by Beosin in no way provides investment advice on any project, nor should it be utilized as investment suggestions of any type. This report represents an extensive evaluation process designed to help our customers improve code quality while mitigating the high risks in blockchain.

3.4 About Beosin

Beosin is the first institution in the world specializing in the construction of blockchain security ecosystem. The core team members are all professors, postdocs, PhDs, and Internet elites from world-renowned academic institutions. Beosin has more than 20 years of research in formal verification technology, trusted computing, mobile security and kernel security, with overseas experience in studying and collaborating in project research at well-known universities. Through the security audit and defense deployment of more than 2,000 smart contracts, over 50 public blockchains and wallets, and nearly 100 exchanges worldwide, Beosin has accumulated rich experience in security attack and defense of the blockchain field, and has developed several security products specifically for blockchain.

BEOSIN
Web3 Security & Compliance

Official Website
https://www.beosin.com

Telegram
https://t.me/beosin

X
https://x.com/Beosin_com

Email
service@beosin.com

LinkedIn
https://www.linkedin.com/company/beosin/