BEOSIN

Blockchain Security

Stable-swap

Smart Contract Security Audit

No. 202309291700

Sep 29th, 2023

SECURING BLOCKCHAIN ECOSYSTEM

Contents

Summary of Audit Results

After auditing, 1 Low-risk and 1 Info-risk items were identified in the Stable-swap 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 : 0	Acknowledged: 1
Info	Fixed : 0	Acknowledged: 1

- **Risk Description:**

Centralization risk, contract administrators can modify key parameters, which in severe cases can affect the user's assets.

● Project Description:

1. Business overview

The Stable-swap project in this audit mainly includes two modules: stable-pool and stable-asset. It mainly implements a decentralized exchange, using the Stable Math algorithm (based on Curve's widely promoted StableSwap), which allows users to freely choose a trading pool for adding liquidity, redeeming, exchanging and other operations.

This audit of the Stable-swap and stable-asset modules in the project. It mainly implements a decentralized exchange, where users are free to choose the pool for adding liquidity, redeeming, exchanging and other operations.

`add_liquidity`: Send the specified tokens from the pool to the contract to get BLP tokens

Redemption includes three different redemption methods:

`redeem_proportion`: Specify the number of BLP tokens to be redeemed and proportionally redeem all types of tokens in the pool.

`redeem_single`: Specify the number of BLP tokens to be redeemed and redeem the single tokens.

`redeem_multi`: Specify the number of tokens to be redeemed and calculate the number of BLP tokens to be redeemed based on the current condition of the pool.

`swap`: send tokens specified in the pool to the contract and swap them for tokens in another pool.

Users incur a fee when trading, which is reflected in a reduction in the amount of money they receive. For example, the amount of BLP gained in `add_liquidity` and the amount exchanged in swap is the amount after deduction of the fee.

The swap fee is calculated in `get_swap_amount`. If the pool's swap_fee is greater than 0, the fee will be deducted from the number of tokens exchanged by the user.

The project's trading pool can only be created by the administrator through the `create_pool` function, and after creation, can call `edit_token_rate` to set the token exchange ratio. The handling fee and the handling fee acceptance address can be modified via `modify_fees` and `modify_recipients`.

Reference Link: https://docs.bifrost.finance/quick-start/vtoken-swap/bifrost-stable-swap

1 Overview

1.1 Project Overview

Project Name	Stable-swap
Project language	Rust
Platform	Polkadot
Github link	https://github.com/bifrost-finance/bifrost/tree/v0.9.82-audit
Audit scope	./pallets/Stable-swap ./pallets/stable-asset (Only key functions `get_pending_fee_amount`, `get_balance_update_amount`, `get_best_route`)
Commit Hash	73af1b5ffc04cf50fe97149e303d8028128f211a

1.2 Audit Overview

Audit work duration: Sep 25, 2023 – Sep 29, 2023

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
Stable-swap-01	Centralization risk	Low	Acknowledged
Stable-swap-02	Redundant code	Info	Acknowledged

Finding Details:

[Stable-swap-01] Centralization risk

Severity Level	Low
Type	Business Security
Lines	Stable-swap/lib.rs L300-310

Description

Centralization risk, the administrators of the contract can modify the handling fee ratio and other key parameters, affecting swap, add liquidity, redeem and other key functions.

For example, the `edit_token_rate` function can modify the conversion ratio between the number of tokens in the pool and their value. In extreme cases, administrators can modify it so that the pool determines that the number of tokens currently available is 0, making it impossible for users to swap or redeem.

```
        pub fn edit_token_rate(
            origin: OriginFor<T>,
            pool_id: StableAssetPoolId,
            token_rate_info: Vec<(
                AssetIdOf<T>,
                (AtLeast64BitUnsignedOf<T>,
AtLeast64BitUnsignedOf<T>),
            )>,
        ) -> DispatchResult {
            T::ControlOrigin::ensure_origin(origin)?;

nutsfinance_stable_asset::Pallet::<T>::set_token_rate(pool_id,
token_rate_info)
        }
```

Recommendation

It is recommended to use a multi-signature wallet or DAO, etc. to manage the administrator rights of the contract and refer to the following two points when modifying key parameters:

1. Regarding the modification of the fee, it is recommended to set a cap on the

	maximum fee. 2. Regarding `modify_a` and `edit_token_rate`, since modifying them involves core price calculations and mathematical models, it is recommended that full auditing and testing be performed prior to modification to reduce potential vulnerabilities and errors.
Status	**Acknowledged.**

[Stable-swap-02] Redundant code

Severity Level Info

Type Coding Conventions

Lines Stable-swap/lib.rs L617-624

Description The redundant code, amounts in the `redeem_single_inner` function, is not used in subsequent calculations and is not reflected in event triggers.

```rust
        let mut amounts: Vec<T::Balance> = Vec::new();
        for idx in 0..pool_size {
            if idx == i_usize {
                amounts.push(dy);
            } else {
                amounts.push(Zero::zero());
            }
        }
```

Recommendation It is recommended to delete the redundant code.

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.

Impact / Likelihood	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

- **Severe**

Severe 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.

3.1.4 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.

3.1.5 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	Deprecated Items
		Redundant Code
		Interface Specification
		Cycles Consumption
2	General Vulnerability	Integer Overflow/Underflow
		Pseudo-random Number Generator (PRNG)
		Transaction-Ordering Dependence
		DoS (Denial of Service)
		Function Call Permissions
		Returned Value Security
		Rollback Risk
		Replay Attack
		Canister Storage Security
		Upgrade Risks
		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
Blockchain Security

Official Website
https://www.beosin.com

Telegram
https://t.me/beosin

Twitter
https://twitter.com/Beosin_com

Email
service@beosin.com