BEOSIN
Web3 Security & Compliance

BetaStaking

Smart Contract Security Audit

No. 202410301141

Oct 30th, 2024

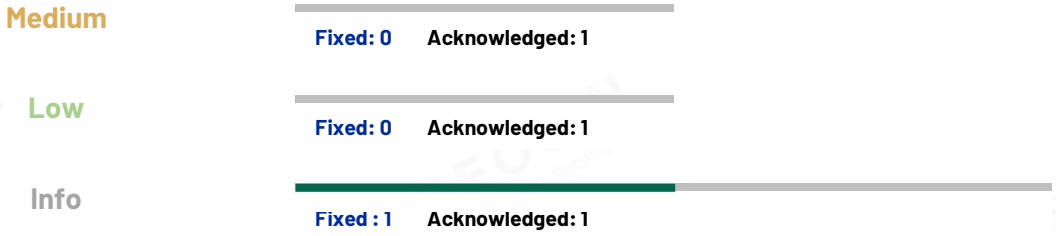
SECURING BLOCKCHAIN ECOSYSTEM

WWW.BEOSIN.COM

Contents

Summary of Audit Results

After auditing, 1 Medium-risk and 1 Low-risk, 2 info item were identified in the BetaStaking project. Specific audit details will be presented in the Findings section. Users should pay attention to the following aspects when interacting with this project:



- **Risk Description:**

If a user only withdraws a portion of their tokens and subsequently calls unstake again, the lock-up period will be updated. This means that tokens that were previously released after the lock-up period may be locked again.However, the project team stated that the frontend will provide relevant risk warnings.

Multiple operators triggering the `unbond` function could postpone withdrawals in `beta_staking`.However, the project team stated that there are protective mechanisms such as an off-chain fund buffer to prevent such situations.

There are flaws in the reward calculation, which may be susceptible to sandwich attacks. However, the project team stated that the off-chain backend system has relevant protective mechanisms to ensure security.

- **Project Description:**

BetaStaking is a staking project that primarily consists of two programs, which will be detailed below.
beta_staking.aleo: This program serves as a user staking pool where users can call the stake_public and stake_private functions to stake credits. These two methods correspond to Aleo's two account models: Account Model and Record Model. After staking, users receive a corresponding amount of the token stALEO. Subsequently, users can submit un-staking requests through two methods: unstake_token, which allows un-staking based on a specific share amount, and unstake_aleo, which allows un-staking based on the desired amount of tokens. After waiting for a certain lock-up period, users can withdraw their funds, with options depending on the account model: withdraw corresponds to the Account Model, while withdraw_private corresponds to the Record Model. Additionally, the token stALEO can be traded. Staking rewards are manually transferred in by whitelisted addresses calling the notify_reward function, and these whitelisted addresses can also call the pull_aleo function to withdraw users' staked funds.
staker.aleo: Through this program, the project team utilizes users' staked funds for validator delegation and reward acquisition. Operators can call the bond function to bond users' staked funds to a validator for delegation. They can later unbond by calling the unbond function. After waiting for a certain lock-up period, operators can call either claim_and_push or the separate claim and push_aleo functions to withdraw and transfer funds back to the user staking pool.

1 Overview

1.1 Project Overview

Project Name	BetaStaking
Project Language	leo
Platform	Aleo
Code Base	https://github.com/BetaStaking-labs/BetaStaking-programs-internal/tree/stake-ext
Commit Hash	dc6131adc1d48f6f7d4874680481308c872dbe43

1.2 Audit Overview

Audit work duration: Oct 25, 2024 – Oct 30, 2024

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 function 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
BetaStaking-01	The reward calculation may cause errors	Medium	Acknowledged
BetaStaking-02	The unbond function may affect user withdrawals	Low	Acknowledged
BetaStaking-03	Logical errors may result in permanent failure of reward updates	Info	Fixed
BetaStaking-04	Redundant code	Info	Acknowledged

Finding Details:

[BetaStaking-01] The reward calculation may have flaws

Severity Level	Medium
Lines	beta_staking.aleo #L408 - 437
Type	Business Security
Description	Reward calculation is independent of time. User can stake a large amount of funds before the project team updates the rewards. After calls the `notify_reward` function to update the rewards, the user can unstake, allowing them to receive a significant portion of the accumulated rewards in the program.
Recommendation	It is recommended that the project team strictly control the amount of rewards updated in a single instance, for example, by establishing a buffer fund pool, and closely monitor on-chain funds.
Status	**Acknowledged.** The project team claims that they will control the amount of rewards updated each time and adjust the calling frequency based on the staking amount. The backend will also monitor on-chain activities, and if any exploitation of this time difference for profit is detected, they will immediately adjust the backend calling strategy to address the issue.

[BetaStaking-02] The unbond function may affect user withdrawals

Severity Level	Low
Lines	staker.aleo #L46 - 56
Type	Business Security
Description	In `staker.aleo`, multiple operators can trigger the program's `unbond` function. This may cause users to experience delays in withdrawals during the lock-up period, as multiple operators triggering the `unbond` function could postpone withdrawals in `beta_staking`.
Recommendation	It is recommended that the project team make modifications based on business logic. If there is a strict off-chain reward calculation function that can determine the `unbond` function for each withdrawal, then if `unbonding` is in progress, further calls to `unbond` should not be allowed. Otherwise, it is advisable for the project team to strictly control the permissions of operators.
Status	**Acknowledged.** The project team claims that they disallows calling `unbond` again if there is an existing `unbonding`. Additionally, they will dynamically allocate a portion of the funds based on recent statistical data to avoid redemption delays in special circumstances.

[BetaStaking-03] Logical errors may result in permanent failure of reward updates

Severity Level	Info
Lines	beta_staking.aleo#L453 - 473
Type	Business Security
Description	The admin can use the `update_settings` function to determine whether the valid status of `stakers` can be modified, with the key point being that a non-modifiable state cannot be switched to a modifiable state. However, this function does not verify whether there are any valid stakers in the current contract during the switch. This may result in the contract being switched to a non-modifiable state when there are no valid stakers present, at which point the contract will no longer be able to perform reward updates or withdrawal operations, and it cannot be fixed afterward.
Recommendation	It is recommended that the project team verify the existence of valid stakers when switching states.
Status	**Fixed.** The program has already been deployed on-chain, and has called the `update_settings` function to switch states, making it impossible to modify the stakers.

[BetaStaking-04] Redundant code

Severity Level	Info
Lines	staker.aleo #L58 - 68
Type	Coding Conventions
Description	The intention of the claim in this contract is to restrict only the operators from initiating the `claim_unbond_public` call. However, this may not achieve the intended effect of the contract, as non-operatorss can still manually invoke `credit.aleo/claim_unbond_public` directly.
Recommendation	It is recommended to remove the redundant permission variables.
Status	**Acknowledged.** The project team claims that they will remove the relevant code in future versions.

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	Redundant Code
		Program naming conflict
		Gas Consumption
		Deprecated Items
2	General Vulnerability	Overflow/Underflow
		Ternary operation safety
		Type conversion safety
		DoS (Denial of Service)
		Function Call Permissions
		Asynchronous call safety
		Returned Value Security
		self.signer Usage
		Record Usage
		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, program developed in Leo language should fix the program naming conflict 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 program itself, such as 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/