BEOSIN
Web3 Security & Compliance

Audiera

Smart Contract Security Audit

No. 202509031659

Sep 3rd, 2025

SECURING BLOCKCHAIN ECOSYSTEM

WWW.BEOSIN.COM

Contents

Summary of Audit Results

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

1. Basic Token Information

Token name	Beat Token
Token symbol	Beat
Decimals	18
Max supply	1 billion
Token type	BEP-20

Table 1 Beat token info

2. Business Description:

This audit primarily covers the following four contracts:

Beat: A BEP-20 standard token with 18 decimals and a maximum total supply of 1 billion. Minting is controlled by the contract owner.

VeBeat: A non-transferable governance token. Minting and burning are controlled by designated authControllers.

VeBeatStaking: A staking contract for Beat tokens. Users earn VeBeat tokens based on staking duration, subject to a maximum cap. VeBeat tokens can be burned to redeem Beat tokens. Notably, even partial Beat withdrawals result in burning all VeBeat tokens, although the user can still claim the remaining Beat balance. The contract also features an accelerated reward mechanism, with related parameters configurable by the owner.

AirdropBeat: An airdrop contract where users with a valid signer's signature can claim Beat tokens through standard or referral-based claims. Referral claims provide additional rewards, and another extra bonuses are distributed based on block timestamps.

1 Overview

1.1 Project Overview

Project Name	Audiera
Project language	Solidity
Platform	BNB Smart Chain
Github Link	https://github.com/Audition2049/audiera-protocol
Commit	23af4b81adbd3ca89dc06814f6efa7823e9753c9 5662af96d122b773e244559505b9a1a44b0663e7
Audit Scope	AirdropBeat.sol VeBeatStaking.sol VeBeat.sol Beat.sol

1.2 Audit Overview

Audit work duration:Sep 2, 2025, Sep 3, 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
Audiera-01	Centralized Risk	Medium	**Fixed**
Audiera-02	Insecure random numbers	Low	**Partially Fixed**
Audiera-03	Signature replay risk	Low	**Fixed**
Audiera-04	Missing events	Info	**Partially Fixed**

Finding Details:

[Audiera-01] Centralized Risk

Severity Level	Medium
Type	Business Security
Lines	VeBeatStaking.sol#L124-151

Description	The owner of the staking contract has the ability to modify reward-related parameters. Additionally, if the Beat token address for staking is changed during the process, it may result in discrepancies in the value of users' deposited assets, potentially causing financial losses.
	`function setBeat(address _beat) external onlyOwner {`
	`require(_beat != address(0));`
	`beat = IERC20(_beat);`
	`}`

Recommendation	It is recommended to delete the function that modifies the beat token and use a multi-signature wallet to manage owner permissions.
Status	**Fixed.** The setBeat function has been removed.

[Audiera-02] Insecure random numbers

Severity Level	Low
Type	Business Security
Lines	AirdropBeat.sol#L210
Description	During the calculation of `drawAmount`, the current time (`curTime`) is used with a modulo operation on gap. Since the current time can be manipulated, this allows `drawAmount` to be maximized arbitrarily. `drawAmount = curTime % gap + minDrawAmount;`
Recommendation	It is recommended to use Chainlink VRF for generating randomness, or alternatively, modify the current algorithm to ensure unpredictability.

Status	**Partially Fixed.** Slightly increased the random number difficulty, considering the small amount of rewards and gas costs, without using external oracles.
	`uint random = uint(blockhash(block.number - 1));`
	`drawAmount = random % gap + minDrawAmount;`

[Audiera-03] Signature replay risk

Severity Level	Low
Type	Business Security
Lines	AirdropBeat.sol#L25-35

Description	The signature information does not contain information such as nonce and chainId. Although reuse is avoided through subsequent state variable checks, there is still a risk of the signature being replayed on other chains.
	`struct ClaimSignDetail { uint256 rewards; address claimer; }`
	`require(isSignatureValid(_sigDetail, keccak256(abi.encode(_detail)), authSigner), 'Signature error');`

Recommendation	It is recommended to add nonce and chainId to the signature.

Status	**Fixed.** Added chainId related checks.
	`struct ClaimSignDetail { uint256 rewards; address claimer; uint256 chainId; }`
	`require(isSignatureValid(_sigDetail, keccak256(abi.encode(_detail)), authSigner), 'Signature error');`
	`require(_detail.chainId == block.chainid, "Invalid network");`

[Audiera-04] Missing events

Severity Level	Info
Type	General Vulnerability
Lines	AirdropBeat.sol#L59-93
Description	The owner modified certain critical parameters without triggering an event. `function setMinBalance(uint256 _bal) external onlyOwner { minBalance = _bal; } function setAuthControllers(address _controller, bool _enable) external onlyOwner { authControllers[_controller] = _enable; }`
Recommendation	It is recommended to add corresponding events and trigger them.
Status	**Partially Fixed.** Added some events for setting functions.

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 Critical 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	Deprecated Items
		Redundant Code
		require/assert Usage
		Default Values
2	General Vulnerability	Insufficient Address Validation
		Lack Of Address Normalization
		Variable Override
		DoS (Denial Of Service)
		Function Call Permissions
		Call/Delegatecall Security
		Tx.origin Usage
		Returned Value Security
		Mathematical Risk
		Overriding Variables
3	Business Security	Business Logics
		Business Implementations
		Manipulable Token Price
		Centralized Asset Control
		Arbitrage Attack
		Access Control

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 Rust 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.

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/