BEOSIN
Web3 Security & Compliance

BJustCoin

Smart Contract Security Audit

No. 202503181112

Mar 18th, 2025

SECURING BLOCKCHAIN ECOSYSTEM

WWW.BEOSIN.COM

Contents

Summary of Audit Results

After the audit, no critical or high-risk vulnerabilities were found in the BJustCoin project. The identified risks include 1 Medium-risk, 2 Low-risks, and 1 Info item. Specific audit details will be presented in the Findings section. Users should pay attention to the following aspects when interacting with this project:

Medium	Fixed : 0 Acknowledged: 1
Low	Fixed : 0 Acknowledged: 2
Info	Fixed : 0 Acknowledged: 1

- **Project Description:**

Business overview

Bjustcoin consists of the following contracts: Bjustcoin, ICOManager, IVestingToken, Oracle, VestingManager, and VestingToken. The following sections provide a detailed introduction to each contract.

Bjustcoin: This contract implements the ERC20 token functionality and includes blacklist management controlled by the owner. The total token supply is 100 million. The owner has the ability to burn tokens from their own account, with permissions managed by the ICOManager contract. Accounts added to the blacklist will be restricted from transferring tokens.

ICOManager: This contract is responsible for token distribution. Upon deployment, it creates the Bjustcoin token and the `VestingManager` contract, which manages token allocations. The contract holds the entire supply of Bjustcoin for sale.

In the allocation plan, the `Strategic`, `Advisors`, `Team`, `FutureTeam`, `Incentives`, `Liquidity`, `Ecosystem`, and `Loyalty` categories have fixed allocations upon contract deployment, collectively accounting for 65% of the total supply. Users can purchase these tokens immediately but must be added to the corresponding whitelist, and the purchase quantity cannot exceed the whitelist limit.

Additionally, the contract supports staged token sales, including `Seed`, `PrivateSale`, `IDO`, and `PublicSale`, which unlock gradually and account for 35% of the total supply. Users must transfer platform tokens to purchase Bjustcoin, with different phases and participants having different token prices. Each phase requires whitelist approval and has purchase limits. Users can only buy tokens from the current phase, and purchases from the next phase will only be available once that phase unlocks.

For initial token distribution, the owner can withdraw or burn unsold tokens. The owner can also carry over unsold tokens from a previous phase to the next phase for sale. Once the contract reaches the EndICO state, all remaining unsold tokens will be burned.

Oracle: Provides the exchange rate when users purchase tokens. The contract verifies the timeliness of the information and ensures that the quoted rate remains within an acceptable fluctuation range.

VestingManager: Creates token lockup contracts for each allocation plan. When users purchase Bjustcoin, the tokens are transferred to these contracts for locking.

VestingToken: Upon receiving staked Bjustcoin, the contract mints an equivalent amount of `VestingToken` for the user. After the user stakes for a specified lockup period, the contract calculates the `unlockable` amount based on the configured schedule. Once the user has a sufficient unlocked balance, they can withdraw Bjustcoin.

1 Overview

1.1 Project Overview

Project Name	BJustCoin
Project Language	Solidity
Platform	Ethereum
Code Base	https://github.com/BJustCoin/BJustCoin
Commit Hash	b53ef9aa8e17af87c1850796db6b09b4e18b75fc

1.2 Audit Overview

Audit work duration: Mar 10, 2025 – Mar 18, 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 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
BJustCoin-01	Centralization Risk	Medium	Acknowledged
BJustCoin-02	ICOManager's Irrevocable Authority Poses Centralization Risk	Low	Acknowledged
BJustCoin-03	Redundant Validation	Low	Acknowledged
BJustCoin-04	Missing Event Triggers in Functions	Info	Acknowledged

Finding Details:

[BJustCoin-01] Centralization Risk

Severity Level	Medium
Lines	Bjustcoin.sol #L 31-36 ICOManager.sol #L 417-423
Type	Business Security

Description	After purchasing and locking tokens, users can obtain Bjustcoin. However, in the Bjustcoin contract implementation, the project owner's owner role has the authority to arbitrarily blacklist user accounts. This introduces a degree of centralization risk.
	`function blacklist(address _address, bool _isBlacklisting) external onlyOwner {`
	`if (blacklists[_address] != _isBlacklisting) {`
	`emit Blacklist(_address, _isBlacklisting);`
	`blacklists[_address] = _isBlacklisting;`
	`}`
	`}`

Recommendation	It is recommended to use a multi-signature wallet to manage related permissions, reducing the risk of unilateral control and enhancing security and decentralization.
Status	**Acknowledged.** The project team stated that they will use a DAO or a multisig wallet to manage the relevant permissions in the future.

[BJustCoin-02] ICOManager's Irrevocable Authority Poses Centralization Risk

Severity Level	Low
Lines	ICOManager.sol #L 520-525
Type	Business Security
Description	Once users hold a certain amount of Bjustcoin, they can transfer tokens to the contract. Since the contract only burns a predetermined fixed amount, any additional tokens sent by users will remain in the contract, preventing its balance from ever reaching zero. This could result in the contract being unable to renounce its authority.
Recommendation	It is recommended to optimize the relevant checks to allow the owner to properly renounce their permissions.
Status	**Acknowledged.** The project team stated that they will manually clear the balance and notify the community accordingly.

```
function bjustCoinTransferOwnership() external onlyOwner {
    if (_baseToken.balanceOf(address(this)) > 0) {
        revert BalanceNotNull();
    }
    _baseToken.transferOwnership(owner());
}
```

[BJustCoin-03] Redundant Validation

Severity Level	Info
Lines	Bjustcoin.sol #L 63-67
Type	Business Security

Description	There is redundant validation of the blacklist status in the contract's transferFrom function, as it has already been implemented in the _update function.
	`function transferFrom(address from, address to, uint256 value) public virtual override returns (bool) {`
	`if (blacklists[to] || blacklists[from]) revert Blacklisted();`

Recommendation	It is recommended to remove the redundant validation in the transferFrom function.
Status	**Acknowledged.**

[BJustCoin-04] Missing the mechanism to trigger the corresponding events

Severity Level	Info
Lines	ICOManager.sol #L 471-474
Type	Coding Conventions

Description	The following functions lack corresponding event triggers when modifying critical variables.
	`function setDefaultRate(uint256 value) external payable onlyOwner {`
	`if (value == 0) revert SetDefaultRateByZero();`
	`defaultRate = value;`
	`}`

Recommendation	It is recommended to add relevant events and trigger them within the corresponding functions.
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	Compiler Version Security
		Deprecated Items
		Redundant Code
		require/assert Usage
		Gas Consumption
2	General Vulnerability	Integer Overflow/Underflow
		Reentrancy
		Pseudo-random Number Generator (PRNG)
		Transaction-Ordering Dependence
		DoS (Denial of Service)
		Function Call Permissions
		call/delegatecall Security
		Returned Value Security
		tx.origin 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, 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/