

atlas

Smart Contract Security Audit

No. 202606171520

Jun 17th, 2026



Contents

1 Overview	6
1.1 Project Overview	6
1.2 Audit Overview	6
1.3 Audit Method	6
2 Findings	8
[atlas-01] Centralization Risk	9
3 Appendix	10
3.1 Vulnerability Assessment Metrics and Status in Smart Contracts	10
3.2 Audit Categories	13
3.3 Disclaimer	15
3.4 About Beosin	16

Summary of Audit Results

After auditing, 1 Low-risk were identified in the atlas 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

● Project Description:

Atlas is a staking and yield-farming project built on PancakeSwap V3. This audit focuses primarily on the following three business contracts—DailyFlow, LockupFlow, Distrib and Transport—each of which is described in turn below.

DailyFlow offers a 200-day staking product that releases rewards linearly on a daily basis. After a user deposits tokens via `lockup`, the contract automatically assigns the order to one of two tiers (Core or Elite, divided at the 2000 threshold), each carrying a different daily reward rate, and locks in the order's principal, start/end times, and projected total rewards at the moment of creation. The user may subsequently call `claim` to collect the rewards accrued over the days elapsed; the contract computes the payable amount based on the actual number of calendar days passed (capped at 200 days), deducts the platform fee `platformFee`, and pays the remainder to the user, while the fee and any surplus yield generated by the liquidity position are funneled to treasury. The contract also exposes query interfaces such as `getOrder` and `getUserOrders`, along with administrative functions—controlled by the project team—for adjusting the fee, the treasury account, and the active position.

LockupFlow offers a fixed-term product that pays principal plus interest in a single settlement at maturity, complementing the day-by-day release model of `DailyFlow`. When a user places an order through `lockup`, they select a lockup tier (`LockupFlowTier`, ranging from a 10-minute test tier up to the 30-day `Imperium` tier across six levels, where longer terms carry higher reward rates); the contract then derives the maturity time from the tier and fixes the principal, earned rewards, and net payout (after fees) in a single step. Funds cannot be withdrawn during the lockup period—only upon maturity may the user claim; at redemption, the contract verifies that the order has not already been claimed (`notClaimed`) and that it has matured, then settles principal and interest, deducts the fee, and disburses the balance to the user.

Transport handles the distribution of the platform's `referral/rebate` rewards, whose amounts are settled off-chain and then submitted on-chain by the project team for payout. Its core function, `claimReferral`, is restricted to the contract Owner; the administrator disburses a specified amount to a designated user on their behalf, likewise deducting the platform fee and crediting both the fee and any surplus position yield to treasury. Unlike the other two contracts, Transport maintains no user staking orders or state and serves purely as a funds-disbursement endpoint. All three contracts inherit from `PositionHandler` and share the deposit/withdraw logic such as `_depositPool` and `_withdrawPool`, consolidating business funds into a single-sided (single-token) PancakeV3 liquidity position to achieve centralized custody of funds and aggregation of yield.

Distrib is a token distribution contract: the contract holds an ERC20 token balance (funded by the Transport contract). The owner (backend) calls distribute to pay out tokens to a specified user — at payout, a platform fee is calculated and deducted based on the `platformFee` rate (with precision `PRECISION = 1e25`), transferred to the treasury, and the remainder is sent to the user.

1 Overview

1.1 Project Overview

Project Name	atlas
Project Language	Solidity
Platform	BNB Chain
File Hash(SHA256)	69cac44e294f0fd5a4d17dd7d59685b82b1f7a55050313405212639a8731da54

1.2 Audit Overview

Audit work duration: Jun 15, 2026 – Jun 17, 2026

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
atlas-01	Centralization Risk	Low	Acknowledged

Finding Details:

[atlas-01] Centralization Risk

Severity Level	Low
Lines	PositionHandler.sol#L26-42
Type	Business Security
Description	All three contracts (Transport, DailyFlow, and LockupFlow) share the same liquidity-position NFT held by the project team. Users generate orders by staking their assets into the contracts, but the underlying position NFT remains under the project team's control and is managed entirely by them. During this period, if a private key is leaked, the owner acts maliciously, or keys are mismanaged, an attacker could manipulate the underlying position through that NFT, thereby posing a centralization risk to users' staked assets. <pre> constructor(address token_, uint256 tokenId_) { require(tokenId_ != 0, "Invalid token id"); token = IERC20Metadata(token_); tokenDecimals = token.decimals(); tokenId = tokenId_; </pre>
Recommendation	It is recommended to place the contract owner and its associated privileges under the management of a multisig wallet to eliminate the single point of failure of a single private key.
Status	Acknowledged. The project team confirmed that this behavior is part of the intended business logic and aligns with the project design.

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

- **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 a leading blockchain security and compliance technology company established in 2018. Being focused on blockchain ecosystem security and compliance, it has developed a product matrix including Beosin KYT, Beosin Trace, and Stablecoin Monitor, which have obtained international certifications such as ISO 27001 and SOC 2. Beosin's core products have been applied for over 70 intellectual property rights, and the company has participated in the development of multiple international standards related to blockchain security. It was among the first batch of enterprises selected for the Cyberport Incubation Programme. Its business covers professional code security audit services for blockchain ecosystems, anti-money laundering compliance technology services for exchanges, financial institutions, and payment institutions, and virtual asset tracing and investigation services for law enforcement and regulatory authorities.

As one of the earliest companies to apply formal verification to blockchain security, Beosin offers professional blockchain and smart contract security audit services. Beosin has audited over 4,500 smart contracts and blockchain projects and has become the official security partner for several renowned blockchains, including BNB Chain, TON, Soneium, Manta Network, Sonic SVM, and SOON Network.



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/>