BINENET

Smart Contract Audit Report

BNBChain

IOST StakingDrop Project

V 1.0

SC.a445b8e8cc03

March 18th, 2025

Table Of Content

1 Report Overview ..... - 2 -
2 Asset Management Security Assessment ..... - 3 -
3 Audit Overview ..... - 4 -
3.1 Project Information ..... - 4 -
3.2 Audit Information ..... - 4 -
3.3 External Visibility Analysis ..... - 4 -
3.4 Audit Process ..... - 5 -
4 Security Finding Details ..... - 6 -
4.1 Staking Mining ..... - 6 -
4.2 Unstake Principal ..... - 6 -
4.3 Withdraw Principal Early with Penalty ..... - 7 -
4.4 Claim Rewards and Principal ..... - 8 -
4.5 Withdraw Principal After The Maximum Claim Period ..... - 9 -
5 Audit Categories ..... - 11 -
6 Explanation Of Vulnerability Rating ..... - 13 -
7 Statement ..... - 15 -
8 About Binenet ..... - 16 -

1 Report Overview

Binenet security team have audited the IOST StakingDrop, 0 risks was identified in IOST StakingDrop. users should pay attention to the following aspects when interacting with this project.

Contract Code	Function	Security Level	Status	Fix Result
Stakingdroop.sol	stake	Info	Audited	---
Stakingdroop.sol	unstake	Info	Audited	---
Stakingdroop.sol	earlyWithdraw	Info	Audited	---
Stakingdroop.sol	claim	Info	Audited	---
Stakingdroop.sol	withdrawPrincipalAfterMaxTime	Info	Audited	---

***Risk Description:** The contract enables users to stake tokens and earn rewards over a specified period. It supports early withdrawal with penalties, claiming rewards after the lock period, and withdrawing principal after a maximum claim period.

2 Asset Management Security Assessment

Asset Type	Function	Security Level
User Mortgage Token Assets	stake/unstake/earlyWithdraw/claim/ withdrawPrincipalAfterMaxTime	Info
Users Mortgage Platform Currency Assets	---	---

***Description:** Check the management security of digital currency assets transferred by users in the contract business logic. Observe whether there are security risks that may cause the loss of customer funds, such as the digital currency assets transferred into the contract are incorrectly recorded or transferred out by mistake.

3 Audit Overview

3.1 Project Information

IOST StakingDrop is a DeFi project on the BNBChain.

The contract enables users to stake tokens and earn rewards over a specified period. It supports early withdrawal with penalties, claiming rewards after the lock period, and withdrawing principal after a maximum claim period. It uses the ERC-20 token standard for staking and rewards distribution.

3.2 Audit Information

Project Name	IOST StakingDrop
Platform	BNBChain
Audit Scope	Stakingdrop.sol#SHA256#49d6fab7a303e5338c04e8ab1dde1323f355d69f2dfef824c52f5984687d383e
Website	https://iostbridge.com

3.3 External Visibility Analysis

Function	Visibility	State Change	Modifier	Payable	Description
stake	external	true	beforeLock	---	---
unstake	external	true	beforeLock	---	---
earlyWithdraw	external	true	afterLock	---	---
claim	external	true	afterLock	---	---
withdrawPrincipalAfterMaxTi	external	true	---	---	---

me					
setStakingStartTime	external	true	onlyOwner	---	---
setStakingDuration	external	true	onlyOwner	---	---
withdrawTokens	external	true	onlyOwner	---	---

3.4 Audit Process

Audit time: 2025.3.18 - 2025.3.18

Audit methods: Static Analysis, Dynamic Testing, Typical Case Testing and Manual Review.

Audit team: Binenet Security Team.

4 Security Finding Details

4.1 Staking Mining

Severity Level : Info

Lines : Stakingdrop.sol # L136

Description: This contract function allows users to stake a specified amount of tokens for a set duration, calculates the staking reward multiplier based on the duration, and updates the user's staking information and global staking state accordingly.

```
ftrace | funcSig
136     function stake(
137         uint256 _amount⬆,
138         uint256 _stakingDay⬆
139     ) external beforeLock nonReentrant {
140         require(_amount⬆ > 0, "StakingDrop: amount must be greater than 0");
141         require(
142             _stakingDay⬆ > 0,
143             "StakingDrop: duration must be greater than 0"
144         );
```

Recommendations: Judging based on business logic.

Status : Audited.

Fix Result: ---

4.2 Unstake Principal

Severity Level : Info

Lines : Stakingdrop.sol # L199

Description: This contract function allows users to unstake their tokens before the staking period begins, returning the staked amount to the user and updating the global staking state accordingly.

```
ftrace | funcSig
199     function unstake() external beforeLock nonReentrant {
200         address user = msg.sender;
201         StakingInfo storage userInfo = stakingInfo[user];
202
203         require(
204             userInfo.stakingAmount > 0,
205             "StakingDrop: no active stake found"
206         );
207
208         uint256 amount = userInfo.stakingAmount;
209
```

Recommendations: Judging based on business logic.

Status : Audited.

Fix Result: ---

4.3 Withdraw Principal Early with Penalty

Severity Level : Info

Lines : Stakingdrop.sol # L226

Description: This contract function allows users to withdraw their staked principal early before the staking period ends, but with a penalty applied to their potential rewards. It updates the user's staking information, calculates the penalty and claimable rewards based on the actual staking duration, and transfers the principal and any available rewards to the user.

```
ftrace | funcSig
226     function earlyWithdraw() external afterLock nonReentrant {
227         address user = msg.sender;
228         StakingInfo storage userInfo = stakingInfo[user];
229
230         require(
231             userInfo.stakingAmount > 0,
232             "StakingDrop: no active stake found"
233         );
234         require(
235             !userInfo.claimedPrincipal,
236             "StakingDrop: principal already claimed"
237         );
238
239         uint256 lockEndTime = stakingStartTime + (userInfo.lockDay * 86400);
240         require(
241             block.timestamp < lockEndTime,
242             "StakingDrop: lock period already ended, use claim() instead"
243         );
```

Recommendations: Judging based on business logic.

Status : Audited.

Fix Result: ---

4.4 Claim Rewards and Principal

Severity Level : Info

Lines : Stakingdrop.sol # L309

Description: This contract function allows users to claim their rewards and principal after the staking lock period has ended and within the maximum claim period. It calculates the total reward, net reward, and claimable reward based on the staking duration and user's previous claims, updates the user's staking state, and transfers the combined rewards and principal to the user.

```
ftrace | funcSig
309     function claim() external afterLock nonReentrant {
310         address user = msg.sender;
311         StakingInfo storage userInfo = stakingInfo[user];
312
313         require(
314             userInfo.stakingAmount > 0,
315             "StakingDrop: no active stake found"
316         );
317
318         uint256 lockEndTime = stakingStartTime + (userInfo.lockDay * 86400);
319         require(
320             block.timestamp >= lockEndTime,
321             "StakingDrop: lock period not ended yet"
322         );
323
324         uint256 maxClaimTime = stakingStartTime + MAX_CLAIM_PERIOD;
325         require(
326             block.timestamp <= maxClaimTime,
327             "StakingDrop: claim period expired"
328         );
```

Recommendations: Judging based on business logic.

Status : Audited.

Fix Result: ---

4.5 Withdraw Principal After The Maximum Claim Period

Severity Level : Info

Lines : Stakingdrop.sol # L368

Description: This contract function allows users to withdraw their staked principal after the maximum claim period has expired, provided the principal has not been claimed previously. It checks the conditions, updates the user's staking state to mark the principal as claimed, and transfers the principal amount back to the user.

```
ftrace | funcSig
368     function withdrawPrincipalAfterMaxTime() external nonReentrant {
369         address user = msg.sender;
370         StakingInfo storage userInfo = stakingInfo[user];
371
372         require(
373             userInfo.stakingAmount > 0,
374             "StakingDrop: no active stake found"
375         );
376
377         uint256 maxClaimTime = stakingStartTime + MAX_CLAIM_PERIOD;
378         require(
379             block.timestamp > maxClaimTime,
380             "StakingDrop: maximum claim period not yet passed"
381         );
382
383         require(
384             !userInfo.claimedPrincipal,
385             "StakingDrop: principal already claimed"
386         );
```

Recommendations: Judging based on business logic.

Status : Audited.

Fix Result: ---

5 Audit Categories

Categories	Subitems
Business Security	Transfer token function
	Mint token and burn token vulnerability
	Contract logic function
	Mining pool deposit and withdrawal function
	Reasonableness of agreement amendment
	Functional design
	Dos caused by time
	Insecure oracles and their design
	Deployer private key leak hazard
General Vulnerability	Compiler version security
	Redundant code
	Use of safemath library
	Not recommended encoding
	Use require/assert mistakely
	Fallback function safety
	tx.origin authentication
	Owner permission control
	Gas consumption detection
	Call injection attack
	Low-level function safety
	Additional token vulnerabilities
	Access control
	Numeric overflow detection
	Arithmetic precision error

	Misuse of random number detection
	Unsafe external call
	Variable override
	Uninitialized storage pointer
	Return value call validation
	Transaction order dependent detection
	Timestamp dependent attack
	Denial of service attack detection
	Fake recharge vulnerability detection
	Reentrancy Attack Detection
	Replay attack detection
	Reordering attack detection

6 Explanation Of Vulnerability Rating

Vulnerability Rating	Rating Description
High Risk Vulnerability	Vulnerabilities that can directly cause the loss of token contracts or user funds, such as: overflow、reentrancy 、false recharge， which can cause the value of tokens to be zeroed, or causing false exchanges to lose tokens, or causing losing ETH or tokens, etc; Vulnerabilities that can cause loss of ownership of token contracts, such as: access control flaws of key functions, call injection leading to access control bypass of key functions, etc; Vulnerabilities that can cause token contracts to fail to work properly, such as: denial of service vulnerabilities caused by sending ETH to malicious addresses, and denial of service vulnerabilities caused by gas exhaustion;
Medium Risk Vulnerability	High-risk vulnerabilities that require specific addresses to be triggered, such as overflow that can only be triggered by token contract owners; access control flaws of non-critical functions, logic design flaws that cannot cause direct financial losses, etc;
Low Risk Vulnerability	Vulnerabilities that are difficult to be triggered, vulnerabilities that cause limited harm after triggering, such as overflow vulnerabilities that require a large amount of ETH or tokens to be triggered, vulnerabilities that the attacker cannot directly profit after triggering overflow, and transaction sequence-dependent risks

	triggered by specifying high gas wait;

7 Statement

Binenet only issues this report based on the facts that have occurred or existed before the issue of this report, and assumes corresponding responsibilities for it. For the facts that occurred or existed after the issuance, we cannot judge the security status of the smart contract , and we will not be responsible for it.

This report does not include external contract calls , new types of attacks that may appear in the future, and contract upgrades or tampered codes (with the development of the project side, smart contracts may add new pools, new functional modules, new external contract calls, etc.), does not include front-end security and server security.

The documents and materials provided to us by the information provider as of the date of this report.

Binenet assumes that there is no missing, tampered, deleted or concealed information provided. If the information provided is missing, tampered, deleted, concealed or reflected inconsistent with the actual situation, Binenet shall not be liable for any losses and adverse effects resulting therefrom.

8 About Binenet

Founded in June 2021, Binenet is a dedicated and pure blockchain security company, focusing on accurate, efficient and intelligent blockchain threat detection and response. Committed to providing users with professional products and dedicated services in the field of blockchain security. Business functions cover penetration testing, code auditing, emergency response, on-chain data monitoring, AML anti-money laundering, etc., covering all aspects of blockchain ecosystem security.

Official Website
https://binenet.com

Telegram
https://t.me/binenetxyz

Twitter
https://twitter.com/binenetxyz

E-mail
team@binenet.com