ETHOS
AUDIT

BSCJungle

Audit Report

Jul. 20, 2022

Contents

Executive Summary ... 3

Audit Details ... 3

Methodology ... 3

Contract Details ... 4

Result Summary ... 4

Issues Reported ... 5

Issues Summary ... 5

Detailed Findings ... 5

Code Documentation ... 5

Adherence to Specifications ... 5

Adherence to Best Practices ... 6

BSCJ-0 – Missing zero-address validation ... 6

BSCJ-1 – Block timestamp reliance ... 6

BSCJ-2 – Function initializing state ... 6

BSCJ-3 – Low-level calls ... 7

BSCJ-4 – State variables that could be constant ... 7

Executive Summary

Audit Details

Project Name	BSCJungle
Codebase	https://www.bscscan.com/address/0xdf0dc0b947fae3a986a4343662d22a35ae1a55dc#code
Initial Audit Date	July 20, 2022
Revision Dates	-
Methodology	Manual

Methodology

This audit's objectives are to evaluate:

- Security-related issues
- Code quality
- Relevant documentation
- Adherence to specifications
- Adherence to best practices

This audit examines the possibility of issues existing along the following vectors (but not limited to):

- Single & Cross-Function Reentrancy
- Front Running (Transaction Order Dependence)
- Timestamp dependence
- Integer Overflow and Underflow
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Number rounding errors
- DoS with (Unexpected) Revert
- DoS with Block Gas Limit
- Insufficient gas griefing
- Forcibly sending native currency
- Logical oversights
- Access control
- Centralization of power
- Logic-Specification Contradiction
- Functionality duplication
- Malicious token minting

The code review conducted for this audit follows the following structure:

1. Review of specifications, documentation to assess smart contract functionality
2. Manual, line-by-line review of code
3. Code's adherence to functionality as presented by documentation
4. Automated tool-driven review of smart contract functionality
5. Assess adherence to best practices
6. Provide actionable recommendations

Contract Details

Contract ID	0xdF0DC0b947faE3a986A4343662d22A35ae1a55Dc
Network	BSC
Language	Solidity
Compiler	v0.8.14+commit.80d49f37
Verification Date	Jul. 20, 2022

Result Summary

Ethos' audit of the **BSCJungle** smart contract has concluded with a **PASSING** result, meaning the contract is largely safe from external threats. The initial review identified a number of informational issues and no risk issues. The remaining report includes all issues identified in the initial review, as well as the revised status post resolution by the team where applicable.

- The smart contract is a variant of the 'miner' meta, however, it contains several advancements and modifications
- It allows users to deposit BUSD tokens into the contract which are time-locked on deposit and redistributed to users over time
- The rate of redistributions depends on the length of time that funds are kept in the contract and varies based on the rate of increase of total value locked
- There are various levels of referral bonuses
- There is an 10% fee applied to deposits, 5% going to marketing wallet and 5% taken as a dev fee
- The contract cannot be closed or shut off at any point after deployment

To conclude, this smart contract does what it is designed to and does not contain any backdoors or owner privileges allowing the contract owner to unexpectedly stop or drain the contract.

Issues Reported

Severity	Unresolved	Acknowledged	Resolved
High	0	0	0
Medium	0	0	0
Low	0	0	0
Info	5	0	0

Issues Summary

ID	Title	Severity	Status
BSCJ-0	Missing zero-address validation	Info	Acknowledged
BSCJ-1	Block timestamp reliance	Info	Acknowledged
BSCJ-2	Function initializing state	Info	Acknowledged
BSCJ-3	Low level calls	Info	Acknowledged
BSCJ-4	State variables could be constant	Info	Acknowledged

Detailed Findings

Code Documentation

The code contains sufficient commenting.

Adherence to Specifications

The BSCJungle smart contract adheres to the smart contract functionality described by the project documentation and is in line with its intended usage.

Adherence to Best Practices

The BSCJungle smart contract adheres to the best practices associated with a standard EVM compatible Solidity smart contract.

BSCJ-0 – Missing zero-address validation

Severity: Informational **Status:** Acknowledged

Description: Functions perform address assignments without checking for zero-address first, in instances where a zero-address assignment is not desired.

Risk: Assigning a zero-address to a crucial address variable where it is not desired may be unwanted functionality and should be checked for.

Recommendation: Check that the address is not zero.

Occurrences: BSCjungle.sol#104-105;

BSCJ-1 – Block timestamp reliance

Severity: Informational **Status:** Acknowledged

Description: ***payoutOf*** function uses conditional statements that is a block timestamp comparison.

Dangerous comparisons:

- block.timestamp > time_end (BSCjungle.sol#245)

Risk: Miners can manipulate block.timestamp value to exploit the require statement and contract.

Recommendation: Avoid using block.timestamp for comparison logic.

BSCJ-2 – Function initializing state

Severity: Informational **Status:** Acknowledged

Description: Variables are set pre-construction with a non-constant function or state variable.

Risk: Special care must be taken when initializing state variables from an immediate function call so as not to incorrectly assume the state is initialized.

Recommendation: Remove any initialization of state variables via non-constant state variables or function calls. If variables must be set upon contract deployment, locate initialization in the constructor instead.

Occurrences: PHT._maxTxAmount; PHT._maxWalletToken; PHT.swapThreshold;

BSCJ-3 – Low-level calls

Severity: Informational **Status**: Acknowledged

Description: The use of low-level calls is error-prone.

Low level calls:

- (success) = recipient.call{value: amount}() (BSCjungle.sol#300)
- (success,returndata) = target.call{value: value}(data) (BSCjungle.sol#329)
- (success,returndata) = target.staticcall(data) (BSCjungle.sol#342)
- (success,returndata) = target.delegatecall(data) (BSCjungle.sol#355)

Risk: Low-level calls do not check for code existence or call success.

Recommendation: Avoid low-level calls. Check the call success. If the call is meant for a contract, check for code existence.

BSCJ-4 – State variables that could be constant

Severity: Informational **Status**: Acknowledged

Description: Constant state variables should be declared constant to save gas.

Risk: Gas optimization

Recommendation: Declare state variables as constant.

Occurrences:

- BSCJUNGLE.DEV_FEE (BSCjungle.sol#67)
- BSCJUNGLE.MRK_FEE (BSCjungle.sol#66)
- BSCJUNGLE.TARIF_MAX_DAYS (BSCjungle.sol#69)
- BSCJUNGLE.TARIF_MIN_DAYS (BSCjungle.sol#68)
- BSCJUNGLE.TARIF_PERC_INCREASE (BSCjungle.sol#70)
- BSCJUNGLE.TARIF_STARTING_ROI (BSCjungle.sol#71)