soken

Smart Contract Audit Report

# INSN

# EXECUTIVE OVERVIEW

**1. Audit details**

| | |
|---|---|
| **Name Token** | **INSN** |
| **Platform** | **bscscan** |
| **Language** | **SOLIDITY** |
| **Date** | **23-10-2024** |

**2. Contract address**

0xA0600d65366e16A3c27faEcBb110EB2E150e48E1

**3. Contract url**

https://bscscan.com/address/0xA0600d65366e16A3c27faEcBb110EB2E150e48E1

## 1. IS SOURCE CODE VERIFIED

The contract’s source code is verified.
Source code verification provides transparency for users interacting with smart contracts. Block explorers validate the compiled code with the one on the blockchain. This also gives users a chance to audit the contracts.

## 2. PRESENCE OF MINTING FUNCTION

The contract cannot mint new tokens. The `_mint` functions was not detected in the contracts.
Mint functions are used to create new tokens and transfer them to the user’s/owner’s wallet to whom the tokens are minted. This increases the overall circulation of the tokens.

## 3. PRESENCE OF BURN FUNCTION

The tokens can be burned in this contract.
Burn functions are used to increase the total value of the tokens by decreasing the total supply.

## 4. SOLIDITY PRAGMA VERSION

The contract can be compiled with an older Solidity version.
Pragma versions decide the compiler version with which the contract can be compiled. Having older pragma versions means that the code may be compiled with outdated and vulnerable compiler versions, potentially introducing vulnerabilities and CVEs.

## 5. PROXY-BASED UPGRADABLE CONTRACT

This is not an upgradable contract.
Having upgradeable contracts or proxy patterns allows owners to make changes to the contract’s functions, token circulation, and distribution.

## 6. OWNERS CANNOT BLACKLIST TOKENS OR USERS

Owners cannot blacklist tokens or users.
If the owner of a contract has permission to blacklist users or tokens, all the transactions related to those entities will be halted immediately.

## 7. IS ERC-20 TOKEN

The contract was found to be using ERC-20 token standard.
ERC-20 is the technical standard for fungible tokens that defines a set of properties that makes all the tokens similar in type and value.

## 8. PAUSABLE CONTRACTS

This is not a Pausable contract.
If a contract is pausable, it allows privileged users or owners to halt the execution of certain critical functions of the contract in case malicious transactions are found.

## 9. CRITICAL ADMINISTRATIVE FUNCTIONS

Critical functions that add, update, or delete owner/admin addresses are not detected
These functions control the ownership of the contract and allow privileged users to add, update, or delete owner or administrative addresses. Owners are usually allowed to control all the critical aspects of the contract.

## 10. CONTRACT/TOKEN SELF DESTRUCT

The contract cannot be self-destructed by owners.
`selfdestruct()` is a special function in Solidity that destroys the contract and transfers all the remaining funds to the address specified during the call. This is usually access-control protected.

## 11. ERC20 RACE CONDITION

The contract is vulnerable to ERC-20 approve Race condition vulnerability.
ERC-20 approve function is vulnerable to a frontrunning attack which can be exploited by the token receiver to withdraw more tokens than the allowance. Proper mitigation steps should be implemented to prevent such vulnerabilities.

## 12. USERS WITH TOKEN BALANCE MORE THAN 5%

No addresses contains more than 5% of circulating token supply.. Token distribution plays an important role when controlling the price of an asset.

## 13. OVERPOWERED OWNERS

The contracts have not defined any owner-controlled functions.
Giving too many privileges to the owners via critical functions might put the user's funds at risk if the owners are compromised or if a rug-pulling attack happens.

## 14. NO COOLDOWN CODE TO HALT TRADING OR WORKFLOWS FOUND

The contract does not have a cooldown feature.
Cooldown functions are used to halt trading or other contract workflows for a certain amount of time so as to prevent users from repeatedly executing transactions or buying and selling tokens.

## 15. OWNERS WHITELISTING TOKENS/USERS

Owners can not whitelist tokens or users.
If the owner of a contract has permission to whitelist users or tokens, it’ll be unfair toward other users or the transaction flow may not be executed impartially.

## 16. OWNERS CAN SET/UPDATE FEES

Owners can not set or update Fees in the contract.

## 17. HARDCODED ADDRESSES

The contract was not hardcoding addresses in the code.

## 18. OWNERS UPDATING TOKEN BALANCE

The contract does not have any owner-controlled functions modifying token balances for users or the contract

## 19. FUNCTION RETRIEVING OWNERSHIP

No such functions were found
If this function exists, it is possible for the project owner to regain ownership even after relinquishing it.

## 20. MALICIOUS TYPECASTING OF ADDRESS

Absence of Malicious Typecasting.
The contract is free from any malicious typecasting of addresses from uint160, ensuring that it maintains the integrity of address handling. This absence of risky typecasting methods enhances the contract's security, protecting it from potential exploitation and preserving user trust.

## 21. GAS ABUSE VIA MALICIOUS MINTING

No such functions were found
The approve() function in the detected contract includes a .mint() function call, which is likely designed to manipulate gas usage. This pattern suggests that the contract is engaging in malicious gas abuse, causing users to unknowingly mint gas tokens and bear the financial burden.