Husky Space

Smart security audit

Space Husky

Audit Details

Audited project
SHKY

Deployer address
0xFE131e51BEac81BFf3D118582f357A0a6281E63D

Client contacts:
Space Husky

Block chain
BNB Smart Chain

Project Website:
https://spacehusky.dog/

Background

SpaceHusky was commissioned by SpaceHusky to perform an audit of smart contracts:

The purpose of the audit was to achieve the following:

- Ensure that the smart contract function is intended
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts Details

Token contract details for SpaceHusky contract

Contract name	SpaceHusky
Contract address	0xFE131e51BEac81BFf3D118582f357A0a6281E63D
Total supply	1,000,000,000
Token ticker	SHKY
Decimals	18
Token holders	7
Transactions count	7
Contract’s current owner address	0xA7Fe9C2Fea9Ee755E146A3d6338CaFc44484f58d

2. Contract Overview

2.1 ERC-20 Standard Compliance

- The contract inherits from OpenZeppelin's BEP-20 contract, ensuring compliance with the BEP-20 token standard.
- Implements basic BEP-20 functions such as transfer, approve, and transferFrom.

2.2 Burnable Tokens

- Inherits from OpenZeppelin's BEP20Burnable extension, allowing token holders to burn (destroy) their tokens, thereby reducing the total supply.

2.3 Token Initialization

- The contract is initialized with a total supply of 100 billion SPH tokens, all of which are minted to the deployer’s address.
- Areas of Concern for Race Conditions, Reentrancy, and Cross-Function Race Conditions:

3. Security Analysis

3.1 Compiler Version

- The contract uses Solidity version 0.8.26, which includes built-in protections against integer overflows and underflows, reducing the risk of these vulnerabilities.

3.2 Race Conditions

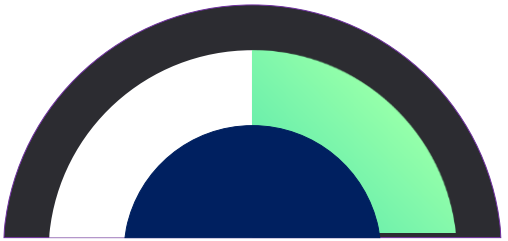
- The contract does not include any complex state-changing functions that could be vulnerable to race conditions. The simplicity of the token transfer and burn functions mitigates these risks.

3.3 Reentrancy

- The contract does not perform any external calls or interact with other contracts during token transfers or burning, minimizing the risk of reentrancy attacks.

3.4 Gas Limit and DoS with Block Gas Limit

- The mint function is executed only once during the contract's deployment, and no additional minting occurs afterward. This prevents potential issues with gas limits during future transactions.

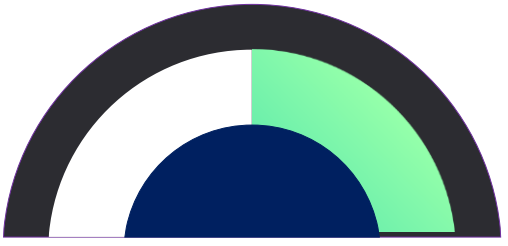


3.5 Ownership and Privileges

- The contract does not include any privileged functions or ownership mechanics. This ensures that no single entity can alter the contract's behavior after deployment.

3.6 External Dependencies

- The contract relies on well-established OpenZeppelin libraries for its BEP-20 and burnable functionalities. These libraries have been thoroughly audited and are widely used in the BINANCE ecosystem.



Issues Checking Status

Issue description	Checking status
1. Compiler errors.	Passed
2. Race conditions and Reentrancy. Cross-function race conditions.	Passed
3 . DoS with block gas limit.	Low issues
4.. Methods execution permissions.	Passed
5. Economy model of the contract.	Passed
6. The impact of the exchange rate on the logic.	Passed
7. Private user data leaks.	Passed
8. Malicious Event log.	Passed
9. Scoping and Declarations.	Passed
10. Uninitialized storage pointers.	Passed
11. Arithmetic accuracy.	Passed
12. Design Logic.	Passed
13. Cross-function race conditions.	Passed
14. Safe Open Zeppelin contracts implementation and usage.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

6. Conclusion

- **The Spacehusky smart contract has been reviewed and found to be secure with no identified vulnerabilities. The contract follows the ERC-20 standard and uses the OpenZeppelin libraries, which are considered reliable and secure. There are no high, medium, or low severity issues detected in this contract, making it safe for deployment.**