verichains

SECURITY AUDIT OF

DRAGONSB SMART CONTRACT

Public Report

Jan 19, 2022

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
Ether (ETH)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
ERC20	ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain.

EXECUTIVE SUMMARY

This Security Audit Report prepared by Verichains Lab on Jan 19, 2022. We would like to thank the DragonSB for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the DragonSB Smart Contract. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified no vulnerable issues in the smart contracts code.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About DragonSB Smart Contract

DragonSB is a Metaverse RPG Game built on the BSC Blockchain where players become a dragon warrior to fight and discover a whole new world - the Dragon world.

In the world of dragons, you can participate in fierce battles, destroy monsters with a superpower to receive SB and NFTs. In addition, you can also transfer NFTs in-game to out-game outside to trading, staking and farming SB to increase earnings.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the DragonSB Smart Contract.

It was conducted on commit 9d449b816611bbb4f504ad7f242f315d1b44630f from git repository *https://github.com/DragonSBFinance/smart-contract*.

1.3. Audit methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 1. Severity levels

1.4. Disclaimer

Please note that security auditing cannot uncover all existing vulnerabilities, and even an audit in which no vulnerabilities are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

2. AUDIT RESULT

2.1. Overview

This table lists some properties of the audited DragonSB Smart Contract (as of the report writing time).

PROPERTY	VALUE
Name	DragonSB
Symbol	SB
Decimals	18
Total Supply	100,000,000 ($x10^{18}$) Note: the number of decimals is 18, so the total representation token will be 100,000,000 or 100 million.

Table 2. The DragonSB Smart Contract properties

2.2. Contract codes

The DragonSB Smart Contract was written in Solidity language, with the required version to be 0.6.12.

2.3. Findings

During the audit process, the audit team found no vulnerability in the given version of DragonSB Smart Contract.

But there are a few things noted below.

2.3.1. Token owner can mint unlimited amount of tokens INFORMATIVE

Token contract contains mint function for onlyOwner so owner of contract can mint unlimited amount of tokens without any cap.

UPDATES

- *Jan 19, 2022*: This issue has been acknowledged and fixed by the DragonSB team. The mint public function was removed.

2.3.2. Version of solidity is too old INFORMATIVE

This project uses Solidity 0.6.12 which was released in July 2020. Latest version of solidity is currently 0.8.9 released on Sep 2021 with various compiler bugs fixed.

RECOMMENDATION

Consider upgrade to the latest version. Solidity 0.8 also integrated SafeMath, so we can remove SafeMath library in case of upgrading.

UPDATES

- *Nov 09, 2021*: This issue has been acknowledged by the DragonSB team.

APPENDIX

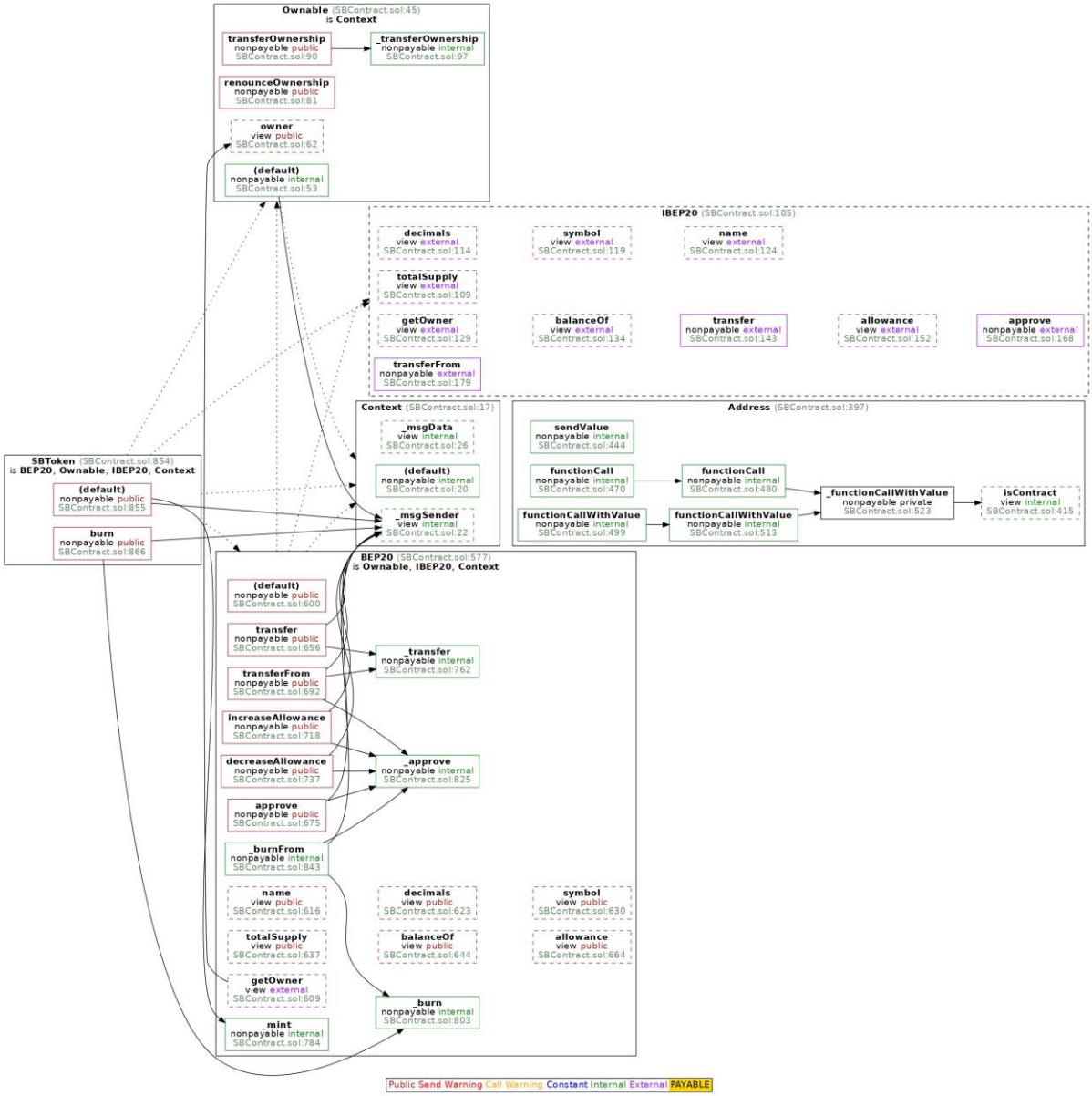

Image 1. DragonSB Smart Contract call graph

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Nov 09, 2021*	Public Report	Verichains Lab
1.1	*Jan 19, 2022*	Public Report	Verichains Lab

Table 3. Report versions history