SECURITY AUDIT OF

BILLIONAIRE PLUS SMART CONTRACT

Public Report

Nov 19, 2021

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 Nov 19, 2021. We would like to thank the BPlus 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 Billionaire Plus 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 Billionaire Plus Smart Contract

Billionaire Plus is an immersive multiplayer board game with 3D and VR modes that incorporates metaverse real estate investment where players own, sell, trade, and collect NFT assets. The game is designed to be fun and competitive, allowing players to own virtual real estate, and customize aspects of the cities within the Billionaire metaverse with owned NFT assets including their character avatars, and buildings, with a marketplace to acquire and trade. It is a journey where players will use digital tokens to enter the game, stake and earn rewards and compete to build an exclusive Dream Lifestyle just like a Billionaire.

BPLUSToken is an ERC20 token that Billionaire Plus players can use in the game.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the smart contracts of Billionaire Plus. It was conducted on commit 0c88df3d6f3077ab7b04cc9aed24d92ca586f1b3

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

The initial review was conducted on Nov 15, 2021 and a total effort of 4 working days was dedicated to identifying and documenting security issues in the code base of the Billionaire Plus Smart Contract.

The following files were made available in the course of the review:

FILE	SHA256 SUM
BPLUSToken.sol	b5b91380b3b4d41dce9a88a632330dc871938ac0ed454cc03f80996e257ff55c
IBEP20.sol	c3f557afb05998444e6d0938ad55114aa34fa3b221cae5dc7bd65c238ba816a5

Table 2 lists some properties of the audited Billionaire Plus Smart Contract (as of the report writing time).

PROPERTY	VALUE
Name	Billionaire Plus Token
Symbol	BPLUS
Decimals	18
Total Supply	10,000,000,000 (x10^{18}) Note: the number of decimals is 18, so the total representation token will be 10,000,000,000 or 10W billion.

Table 2. The Billionaire Plus Smart Contract properties

2.2. Contract codes

The Billionaire Plus Smart Contract was written in Solidity language, with the required version to be 0.8.4.

The contract imported the Context, Ownable, Pausable contract which were implemented by OpenZeppelin. The contract inherits Pausable contract therefore the contract can pause or unpause the activities.

In addition, the contract contains mint function for onlyOwner so owner of contract can mint unlimited amount of tokens without any cap.

2.3. Findings

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

2.4. Additional notes and recommendations

2.4.1. BPLUSToken.sol - Unnecessary usage of SafeMath library in Solidity 0.8.0+ INFORMATIVE

All safe math usage in the contract are for overflow checking, solidity 0.8.0+ already do that by default, the only usage of safemath now is to have a custom revert message which isn't the case in the auditing contracts. We suggest using normal operators for readability and gas saving.

Currently, the methods of safemath are used in in transferFrom, increaseAllowance, decreaseAllowance, _transfer, _mint, _burn and _burnFrom functions.

RECOMMENDATION

We suggest changing all methods from SafeMath library to normal arithmetic operator that we regarded above.

APPENDIX

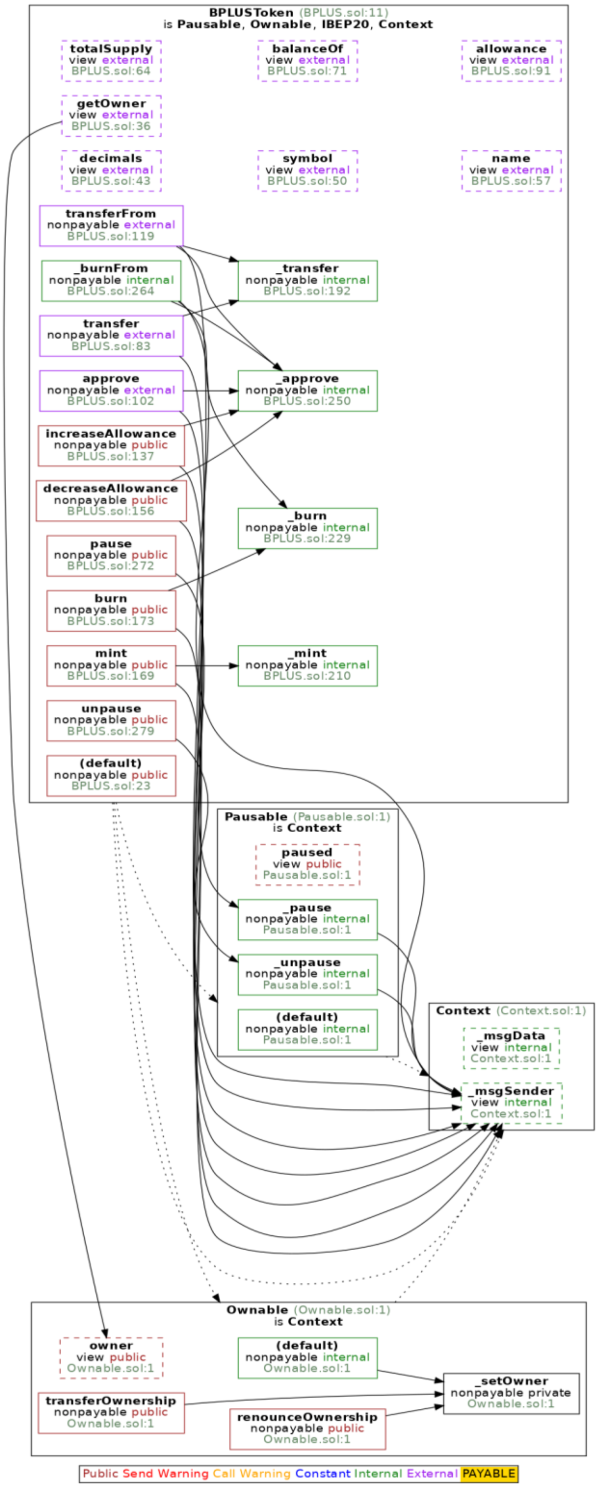
BPLUSToken (BPLUS.sol:11)
is Pausable, Ownable, IBEP20, Context
totalSupply view external BPLUS.sol:64
balanceOf view external BPLUS.sol:71
allowance view external BPLUS.sol:91
getOwner view external BPLUS.sol:36
decimals view external BPLUS.sol:43
symbol view external BPLUS.sol:50
name view external BPLUS.sol:57
transferFrom nonpayable external BPLUS.sol:119
_burnFrom nonpayable internal BPLUS.sol:264
_transfer nonpayable internal BPLUS.sol:192
transfer nonpayable external BPLUS.sol:83
approve nonpayable external BPLUS.sol:102
_approve nonpayable internal BPLUS.sol:250
increaseAllowance nonpayable public BPLUS.sol:137
decreaseAllowance nonpayable public BPLUS.sol:156
_burn nonpayable internal BPLUS.sol:229
pause nonpayable public BPLUS.sol:272
burn nonpayable public BPLUS.sol:173
mint nonpayable public BPLUS.sol:169
_mint nonpayable internal BPLUS.sol:210
unpause nonpayable public BPLUS.sol:279
(default) nonpayable public BPLUS.sol:23
Pausable (Pausable.sol:1)
is Context
paused view public Pausable.sol:1
_pause nonpayable internal Pausable.sol:1
_unpause nonpayable internal Pausable.sol:1
(default) nonpayable internal Pausable.sol:1
Context (Context.sol:1)
_msgData view internal Context.sol:1
_msgSender view internal Context.sol:1
Ownable (Ownable.sol:1)
is Context
owner view public Ownable.sol:1
(default) nonpayable internal Ownable.sol:1
_setOwner nonpayable private Ownable.sol:1
transferOwnership nonpayable public Ownable.sol:1
renounceOwnership nonpayable public Ownable.sol:1
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 1. Billionaire Plus Smart Contract call graph

3. VERSION HISTORY

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

Table 3. Report versions history