HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Bizverse World PTE. LTD
Date: March 7th, 2022

Document

Name	Smart Contract Code Review and Security Analysis Report for Bizverse World PTE. LTD.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	ERC20 token; Vesting
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/BIZVERSE-LAB/smartcontracts-hacken-audit
Commit	a8ff52fb65b60c9f2bac75694662efe3a6c12026
Deployed contract	–
Technical Documentation	YES
JS tests	NO
Website	
Timeline	16 FEBRUARY 2022 – 07 MARCH 2022
Changelog	25 FEBRUARY 2022 – INITIAL AUDIT 07 MARCH 2022 – SECOND REVIEW

Table of contents

Introduction

Hacken OÜ (Consultant) was contracted by Bizverse World PTE. LTD. (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contract and its code review conducted between February 16th, 2022 - March 7th, 2022.

Scope

The scope of the project is smart contracts in the repository:
Repository:
    https://github.com/BIZVERSE-LAB/smartcontracts-hacken-audit
Commit:
    a8ff52fb65b60c9f2bac75694662efe3a6c12026
Technical Documentation: Yes —
https://docs.google.com/document/d/1w6l4-UfFoAnU3ujY7qF4C1TMY4wwlvq5_XnoZ3m
xVDk/edit
JS tests: No
Contracts:
    BIVE/openzeppelin/Context.sol
    BIVE/openzeppelin/ERC20.sol
    BIVE/openzeppelin/ERC20Pausable.sol
    BIVE/openzeppelin/IERC20.sol
    BIVE/openzeppelin/IERC20Metadata.sol
    BIVE/openzeppelin/Ownable.sol
    BIVE/openzeppelin/Pausable.sol
    BIVE/Bizverse.sol
    BIVE/BizverseWorldERC20.sol
    BIVE/Blacklistable.sol
    BIVE/ERC20Blacklistable.sol
    Allocate.sol

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	- Reentrancy - Ownership Takeover - Timestamp Dependence - Gas Limit and Loops - DoS with (Unexpected) Throw - DoS with Block Gas Limit - Transaction-Ordering Dependence - Style guide violation - Costly Loop - ERC20 API violation - Unchecked external call - Unchecked math - Unsafe type inference - Implicit visibility level - Deployment Consistency - Repository Consistency - Data Consistency
Functional review	- Business Logics Review - Functionality Checks - Access Control & Authorization - Escrow manipulation - Token Supply manipulation - Assets integrity - User Balances manipulation - Data Consistency manipulation - Kill-Switch Mechanism - Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contracts are secured.

Insecure	Poor secured	Secured	Well-secured

You are here

Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. All found issues can be found in the Audit overview section.

As a result of the audit, security engineers found **1** medium and **2** low severity issues.

After the second review security engineers found that low issues were resolved and the medium one was commented on. As per the customer's comment, the constructor arguments would be used to provide the correct TGE parameters. However, that could not be checked by the security team unless the contract is deployed. Therefore we still have unresolved **1** medium issue.

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution

Audit overview

■■■■ Critical

No critical issues were found.

■■■ High

No high severity issues were found.

■■ Medium

1. Functionality not implemented.

   While the provided documentation states exact allocations and vesting terms, the implementation of the current contracts don't specify that. Those allow to set up any allocations or vesting terms that could be not related to what is described in docs.

   Contracts: Allocate.sol

Recommendation: Please make sure your contracts have allocation and vesting terms specified in either the constructor or initialization function.

   Status: Acknowledged. Would be using constructor arguments.

■ Low

1. Too many digits.

   Literals with many digits are difficult to read and review.

   Contracts: Bizverse.sol

   Function: constructor

Recommendation: Please use scientific notation or separate by underscores (ie: 1e13, or 1e9 * 10**4, or 1_000_000_000_0000)

   Status: Fixed.

2. External calls in the loop.

   In the for-loop, there is a "transferFrom" call which is burning a lot of gas.

   Contracts: Allocate.sol

   Function: batchLock

Recommendation: Please sum all amounts and do a single transferFrom-call after the loop.

   Status: Fixed.

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools.

The audit report contains all found security vulnerabilities and other issues in the reviewed code.

As a result of the audit, security engineers found **1** medium and **2** low severity issues.

After the second review security engineers found that low issues were resolved and the medium one was commented on. As per the customer's comment, the constructor arguments would be used to provide the correct TGE parameters. However, that could not be checked by the security team unless the contract is deployed. Therefore we still have unresolved **1** medium issue.

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the audit can't guarantee the explicit security of the audited smart contracts.