IBBA Smart Contract Initial Audit Report

Executive Summary

Project Name: **IBBA**
Audited: **Amadasun Goodness**
Overview: **The re-evaluation of the IBBA minting contract is an ERC721 smart contract that batch mints NFTs.**
Timeline: **May 14th–15th, 2023**
Method: **manual review, functional testing, automated testing, etc.**
Audit Scope: **The scope of this audit was to re-evaluate the IBBA minting codebase for quality, security, and correctness.**

Fixed In: **May 14th 2023**

Number of security issues per severity.

TYPE	HIGH	MEDIUM	LOW	INFORMATIONAL
OPEN	0	0	0	0
ACKNOWLEDGED	0	0	0	0
Partially Resolved	0	0	0	0
CLOSED	0	0	0	0

Checked Vulnerabilities

- **Re-entrancy**
- **Timestamp Dependence**
- **Gas Limits, and Loops**
- **DoS with a Block Gas Limit**
- **Transaction-Ordering Dependence**
- **Use of tx.origin**
- **Exception disorder**

- Gasless send
- Balance equality
- Byte array
- Transfer forward all gas
- ERC721 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

The smart contract was checked against the above vulnerabilities, and there was no issue found.

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- **The overall quality of the code.**
- **Use of best practices.**
- **Code documentation and comments match logic and expected behavior.**
- **Implementation of ERC-721 token standards.**
- **efficient use of gas.**
- **Code is safe from re-entrancy and other vulnerabilities.**

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this stage, we have examined the structure and design patterns of smart contracts. The smart contract's structure has been thoroughly examined to make sure it won't cause any issues in the future.

Static Analysis
To find contract weaknesses, a static analysis of smart contracts was conducted. This stage involves testing the security of smart contracts using a number of automated techniques.

Code Review / Manual Analysis
To find additional vulnerabilities or confirm the vulnerabilities discovered during the static analysis, a manual code analysis or review was conducted. The whitepaper's reasoning was validated and contrasted with the analysis of the contracts, which was done entirely manually. Additionally, the outcomes of the automatic analysis were checked by hand.

Gas Consumption
In this stage, we have examined the behavior of smart contracts in production. Checks were made to determine the amount of gas utilized and the potential for code optimization to lower gas usage.

Tools and platforms used for auditing
Remix IDE, Truffle,Solhint, Mythril, Slither, and Solidity statistical analysis.

Types of severity

Every issue in this report has been assigned a severity level. There are four levels of severity, and each of them has been explained below.

High Severity Issues

If there is a high severity problem or vulnerability, your smart contract is vulnerable to attack. These issues must be resolved prior to transitioning to a live environment because they are crucial to the performance or operation of the smart contract.

Medium Severity Issues

The problems classified as "medium severity" are typically brought on by mistakes and flaws in the smart contract code. These kinds of problems should still be solved since they could arise.

Low-severity Severity Issues

Low severity problems might have a negligible effect and are only alerts that can be ignored for the time being. It would be preferable to address these problems in the future.

Types of Issues

Open

Security vulnerabilities were identified that must be resolved and are currently unresolved.

Resolved

These were the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities that have been acknowledged but are yet to be resolved

Partially Resolved

Considerable efforts have been invested to reduce the risk and impact of the security issue, but it has not been completely resolved.

Low Severity Issues

A.1 Floating Pragma

Description: The same compiler version and settings that the contracts have undergone extensive testing with should be used for deployment. Locking the pragma helps protect against the unintentional deployment of contracts using, for instance, an out-of-date compiler version that could bring problems that harm the contract system.

Remediation: Here all the in-scope contracts have an unlocked pragma; it is recommended to use the 0.8.17 version.

Status: RESOLVED

A.2: Add an external modifier instead of public.

Description: It is recommended to use the external access modifier instead of public

for the following functions that are not called from the contract:

- **BatchMinting()**
- **pause()**
- **unpause()**

Status: RESOLVED

Functional Tests

Some of the tests performed are mentioned below:

- ✓ **should be able to deploy and mint the NFT.**
- ✓ **Should be able to mint more NFTs.**
- ✓ **Should be able to transfer NFTs to addresses.**
- ✓ **Should be able to activate and deactivate the contract.**
- ✓ **Should be able to mint NFTs to house wallet**
- ✓ **Should be able to change ownership**
- ✓ **Should be able to retrieve NFT metadata uri**

Closing Summary

In this report, we have considered the security of the IBBA minting contract. We performed our audit according to the procedure described above.

Some issues of low severity were found. Some suggestions and best practices are also provided in order to improve the code quality and security posture.

Disclaimer

The smart contract audit is not an endorsement of the IBBA Platform, a security warranty, or financial advice. The audited smart contracts are not guaranteed to be secure or accurate.

The writers of this book should not be held liable for decisions taken in reliance on the information included herein, and it should not be understood as investment or legal advice. Smart contract security requires several steps. A single audit cannot be deemed sufficient. We advise the IBBA Team to implement a bug reward scheme to promote more examination of the smart contract by other outside parties.