BEOSIN
Blockchain Security

SoulBound Protocol

Smart Contract Security Audit

No. 202312270916

Dec 27th, 2023

SECURING BLOCKCHAIN ECOSYSTEM

WWW.BEOSIN.COM

Contents

Summary of Audit Results

After auditing, 1 High item was identified in the SoulBound Protocol project. Specific audit details will be presented in the **Findings section.** Users should pay attention to the following aspects when interacting with this project:

High
Fixed:1

- **Project Description:**

Business overview

SoulBound is a NFT-based launchpad. And the SoulBound has the functions of NFT buying， selling and canceling sales. Users can conduct NFT transactions through the payment token set when deploying the NFTMarketplace contract.

1 Overview

1.1 Project Overview

Project Name	**SoulBound Protocol**	
Project language	Solidity	
File Hash (SHA256)	marketPlace.sol	7cb26dbb64673fa7f0091e45867bda3f153ea0c68a1888bd8fd3ad6155831cbf bb01334511a838c833965c4b76216d8a93e8f3109eef3a62525f583170b089e4
	ReentrancyGuard.sol	66f741378925c436cf2de216087e4f393f34d33cd9fde35d668b8620e5cba0c4
	StringHelper.sol	857504c68c68ad691a489b892b71bd57a3afd73fb92c35c4c605c88d1427b48a
	ArrayFind.sol	f9d9e602f9591445c2d9a92349f136ef995db5d0e1a9a5dbc7fb65bb7cd760e9
	ArrayRemove.sol	bb8e4c317355e6623d288942e08f1ad2d956f8e76e2bdf4365e29098603d4a99
	Error.sol	cd5feb7d7dce17b816b1ed7116b4c50ed87038bfedcee93c5fdab87edc677180
	IERC20.sol	a265b5f773e7680d485201ba5e5227fdfd6529b4b082130632493bdc57369791

1.2 Audit Overview

Audit work duration: Nov 24, 2023 – Dec 27, 2023

Audit team: Beosin Security Team

1.3 Audit Method

The audit methods are as follows:

1. Formal Verification

Formal verification is a technique that uses property-based approaches for testing and verification. Property specifications define a set of rules using Beosin's library of security expert rules. These rules call into the contracts under analysis and make various assertions about their behavior. The rules of the specification play a crucial role in the analysis. If the rule is violated, a concrete test case is provided to demonstrate the violation.

2. Manual Review

Using manual auditing methods, the code is read line by line to identify potential security issues. This ensures that the contract's execution logic aligns with the client's specifications and intentions, thereby safeguarding the accuracy of the contract's business logic.

The manual audit is divided into three groups to cover the entire auditing process:

The Basic Testing Group is primarily responsible for interpreting the project's code and conducting comprehensive functional testing.

The Simulated Attack Group is responsible for analyzing the audited project based on the collected historical audit vulnerability database and security incident attack models. They identify potential attack vectors and collaborate with the Basic Testing Group to conduct simulated attack tests.

The Expert Analysis Group is responsible for analyzing the overall project design, interactions with third parties, and security risks in the on-chain operational environment. They also conduct a review of the entire audit findings.

3. Static Analysis

Static analysis is a method of examining code during compilation or static analysis to detect issues. Beosin-VaaS can detect more than 100 common smart contract vulnerabilities through static analysis, such as reentrancy and block parameter dependency. It allows early and efficient discovery of problems to improve code quality and security.

2 Findings

Index	Risk description	Severity level	Status
SP-01	ListTokenIds handling exception	High	Fixed

Finding Details:

[SP-01] ListTokenIds handling exception

Severity Level High

Type Business Security

Lines marketPlace.sol #L83-89

Description The cancelListing function did not transfer the user's NFT out, deleted the listing data but did not pop the data in the listedTokenIds, and did not set the NFT list status to false. This will result in:

1. Users cannot withdraw listed NFT;
2. Unable to receive rewards;
3. There is incorrect data in listedTokenIds.

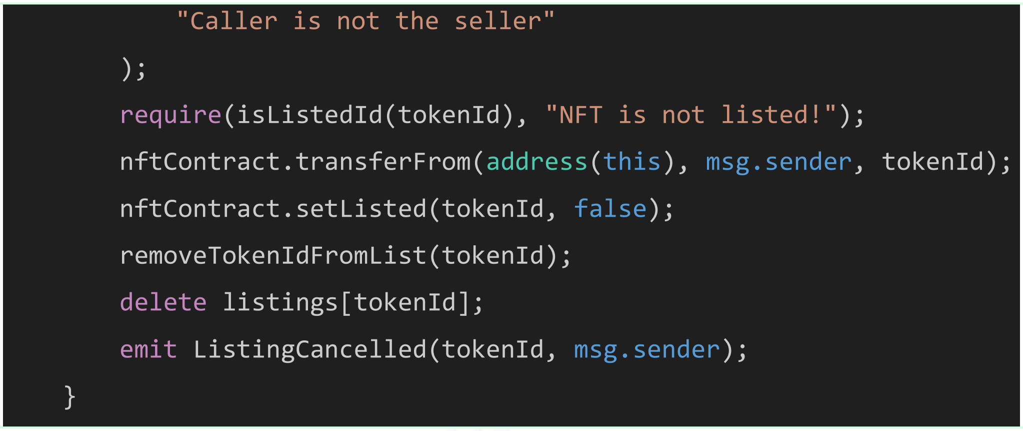
```
    function getAllListedNFTs() external view returns (Listing[] memory) {
        Listing[] memory allListings = new Listing[](listedTokenIds.length);
        for (uint256 i = 0; i < listedTokenIds.length; i++) {
            allListings[i] = listings[listedTokenIds[i]];
        }
        return allListings;
    }

    function cancelListing(uint256 tokenId) external {
        require(listings[tokenId].seller == msg.sender, "Caller is not the seller");
        delete listings[tokenId];
        emit ListingCancelled(tokenId, msg.sender);
    }
```

Recommendation It is recommended to return the NFT to the user, pop out the data in listedTokenIds array and change NFT.listed to false in the cancelListing function.

Status **Fixed.** In the new version of the code, this issue has been fixed according to the modification.

```
    function cancelListing(uint256 tokenId) external {
        require(
            listings[tokenId].seller == msg.sender,
```



```
            "Caller is not the seller"
        );
        require(isListedId(tokenId), "NFT is not listed!");
        nftContract.transferFrom(address(this), msg.sender, tokenId);
        nftContract.setListed(tokenId, false);
        removeTokenIdFromList(tokenId);
        delete listings[tokenId];
        emit ListingCancelled(tokenId, msg.sender);
    }
```

3 Appendix

3.1 Vulnerability Assessment Metrics and Status in Smart Contracts

3.1.1 Metrics

In order to objectively assess the severity level of vulnerabilities in blockchain systems, this report provides detailed assessment metrics for security vulnerabilities in smart contracts with reference to CVSS 3.1(Common Vulnerability Scoring System Ver 3.1).

According to the severity level of vulnerability, the vulnerabilities are classified into four levels: "critical", "high", "medium" and "low". It mainly relies on the degree of impact and likelihood of exploitation of the vulnerability, supplemented by other comprehensive factors to determine of the severity level.

Impact Likelihood	Severe	High	Medium	Low
Probable	Critical	High	Medium	Low
Possible	High	Medium	Medium	Low
Unlikely	Medium	Medium	Low	Info
Rare	Low	Low	Info	Info

3.1.2 Degree of impact

- **Severe**

Severe impact generally refers to the vulnerability can have a serious impact on the confidentiality, integrity, availability of smart contracts or their economic model, which can cause substantial economic losses to the contract business system, large-scale data disruption, loss of authority management, failure of key functions, loss of credibility, or indirectly affect the operation of other smart contracts associated with it and cause substantial losses, as well as other severe and mostly irreversible harm.

- **High**

High impact generally refers to the vulnerability can have a relatively serious impact on the confidentiality, integrity, availability of the smart contract or its economic model, which can cause a greater economic loss, local functional unavailability, loss of credibility and other impact to the contract business system.

- **Medium**

Medium impact generally refers to the vulnerability can have a relatively minor impact on the confidentiality, integrity, availability of the smart contract or its economic model, which can cause a small amount of economic loss to the contract business system, individual business unavailability and other impact.

- **Low**

Low impact generally refers to the vulnerability can have a minor impact on the smart contract, which can pose certain security threat to the contract business system and needs to be improved.

3.1.4 Likelihood of Exploitation

- **Probable**

Probable likelihood generally means that the cost required to exploit the vulnerability is low, with no special exploitation threshold, and the vulnerability can be triggered consistently.

- **Possible**

Possible likelihood generally means that exploiting such vulnerability requires a certain cost, or there are certain conditions for exploitation, and the vulnerability is not easily and consistently triggered.

- **Unlikely**

Unlikely likelihood generally means that the vulnerability requires a high cost, or the exploitation conditions are very demanding and the vulnerability is highly difficult to trigger.

- **Rare**

Rare likelihood generally means that the vulnerability requires an extremely high cost or the conditions for exploitation are extremely difficult to achieve.

3.1.5 Fix Results Status

Status	Description
Fixed	The project party fully fixes a vulnerability.
Partially Fixed	The project party did not fully fix the issue, but only mitigated the issue.
Acknowledged	The project party confirms and chooses to ignore the issue.

3.2 Audit Categories

No.	Categories	Subitems
1	Coding Conventions	Compiler Version Security
		Deprecated Items
		Redundant Code
		require/assert Usage
		Gas Consumption
2	General Vulnerability	Integer Overflow/Underflow
		Reentrancy
		Pseudo-random Number Generator (PRNG)
		Transaction-Ordering Dependence
		DoS (Denial of Service)
		Function Call Permissions
		call/delegatecall Security
		Returned Value Security
		tx.origin Usage
		Replay Attack
		Overriding Variables
		Third-party Protocol Interface Consistency
3	Business Security	Business Logics
		Business Implementations
		Manipulable Token Price
		Centralized Asset Control
		Asset Tradability
		Arbitrage Attack

Beosin classified the security issues of smart contracts into three categories: Coding Conventions, General Vulnerability, Business Security. Their specific definitions are as follows:

- **Coding Conventions**

Audit whether smart contracts follow recommended language security coding practices. For example, smart contracts developed in Solidity language should fix the compiler version and do not use deprecated keywords.

- **General Vulnerability**

General Vulnerability include some common vulnerabilities that may appear in smart contract projects. These vulnerabilities are mainly related to the characteristics of the smart contract itself, such as integer overflow/underflow and denial of service attacks.

- **Business Security**

Business security is mainly related to some issues related to the business realized by each project, and has a relatively strong pertinence. For example, whether the lock-up plan in the code match the white paper, or the flash loan attack caused by the incorrect setting of the price acquisition oracle.

*Note that the project may suffer stake losses due to the integrated third-party protocol. This is not something Beosin can control. Business security requires the participation of the project party. The project party and users need to stay vigilant at all times.

3.3 Disclaimer

The Audit Report issued by Beosin is related to the services agreed in the relevant service agreement. The Project Party or the Served Party (hereinafter referred to as the "Served Party") can only be used within the conditions and scope agreed in the service agreement. Other third parties shall not transmit, disclose, quote, rely on or tamper with the Audit Report issued for any purpose.

The Audit Report issued by Beosin is made solely for the code, and any description, expression or wording contained therein shall not be interpreted as affirmation or confirmation of the project, nor shall any warranty or guarantee be given as to the absolute flawlessness of the code analyzed, the code team, the business model or legal compliance.

The Audit Report issued by Beosin is only based on the code provided by the Served Party and the technology currently available to Beosin. However, due to the technical limitations of any organization, and in the event that the code provided by the Served Party is missing information, tampered with, deleted, hidden or subsequently altered, the audit report may still fail to fully enumerate all the risks.

The Audit Report issued by Beosin in no way provides investment advice on any project, nor should it be utilized as investment suggestions of any type. This report represents an extensive evaluation process designed to help our customers improve code quality while mitigating the high risks in blockchain.

3.4 About Beosin

Beosin is the first institution in the world specializing in the construction of blockchain security ecosystem. The core team members are all professors, postdocs, PhDs, and Internet elites from world-renowned academic institutions. Beosin has more than 20 years of research in formal verification technology, trusted computing, mobile security and kernel security, with overseas experience in studying and collaborating in project research at well-known universities. Through the security audit and defense deployment of more than 2,000 smart contracts, over 50 public blockchains and wallets, and nearly 100 exchanges worldwide, Beosin has accumulated rich experience in security attack and defense of the blockchain field, and has developed several security products specifically for blockchain.

BEOSIN
Blockchain Security

Official Website
https://www.beosin.com

Telegram
https://t.me/beosin

Twitter
https://twitter.com/Beosin_com

Email
service@beosin.com