# LUDO

# SMART CONTRACT AUDIT

## Certification

## LUDO RANK NFT MINTER AND MANAGEMENT LOGIC FOR MULTIVERSX

xaudits

**SMART CONTRACT AUDITS**

Sep 13th, 2024 / v.0.1

Audited source code version:

**352f996a71d1bf05e0265b68659284b07f5883f8**

## Structure and Organization of the Document

Some sections are more important than others. The most critical areas are at the top, and the less critical sections are at the bottom. The issues in these sections have been fixed or addressed and will show by the "Resolved" or "Unresolved" tags. Each case is written so you can understand how serious it is, with an explanation of whether it is a risk of exploitation or unexpected behavior.

CRITICAL

These issues can have a dangerous effect on the ability of the contract to work correctly.

HIGH

These issues significantly affect the ability of the contract to work correctly.

MEDIUM

These issues affect the ability of the contract to operate correctly but do not hinder its behavior.

LOW

These issues have a minimal impact on the contract's ability to operate.

INFORMATIONAL

These issues do not impact the contract's ability to operate.

## Issues

### 1. Wrong event key

**Fixed / INFORMATIONAL**

Description: The key for the **“tier_stayed_event”** is incorrect and appears to have been mistakenly copied from the **“tier_upgraded_event.”**

**! Possible fix to research**

```
Fix the event key to make it different from the upgrade event.
```

**! Response**

```
Fixed.
```

**! Status**

```
Accepted & Closed.
```

### 2. Owner private key is used as signer

**Fixed / INFORMATIONAL**

Description: The owner’s private key is currently being used as the signer. This approach can be problematic, particularly if you plan to deploy the contract from a hardware wallet or ledger. Storing the ledger’s private key on a server could expose it to multiple vulnerabilities.

**! Possible fix to research**

```
A better practice would be to generate a separate private and public key
pair for signing transactions. This dedicated key could be used to sign
transactions and can be updated as needed on both the smart contract (SC)
and the backend signer.
```

**! Response**

```
Fixed.
```

**! Status**

```
Accepted & Closed.
```

## 3. Nonce not checked on mints and upgrades

Fixed / MEDIUM

Description: The nonce value is never read from the smart contract; it is always provided through the transaction parameters and updated from there. However, there is no check to ensure that the nonce given in the parameters is consistent with the one stored on-chain.

**! Possible fix to research**

```
Before updating the nonce in the storage and executing any operation,
check if the nonce sent in the parameters is the same as the one
previously stored.
```

**! Response**

```
Fixed.
```

**! Status**

```
Accepted & Closed.
```

## 4. Contract needs to be payable for integrator updates

Fixed / MEDIUM

Description: The contract should be set as non-payable for safety and security reasons. This approach ensures full control over all asset transfers in and out of the smart contract storage.
Failing to store what an individual user sends could result in the user losing their NFT if the update is not completed by the integrator.

**! Possible fix to research**

```
For the integrator_update_rank endpoint, the proposed logic is as
follows:
a. The integrator calls a new pay_for_update endpoint, providing details
of the user for whom the update should be paid.
b. This information is then stored in the smart contract storage. c. Once
this is completed, the user (who holds the NFT in their wallet) sends
the NFT to the integrator_update_rank endpoint, where the payment is
verified, and the NFT is upgraded. The integrator_update_rank endpoint
should also be updated to receive the user's NFT.
```

**! Response**

```
Fixed.
```

**! Status**

```
Accepted & Closed.
```

## 5. Rank Histories array could cause gas issues

Fixed / LOW

Description: After a large number of elements are stored in the “rank_histories” VecMapper, the gas required to read or interact with the mapper may exceed the maximum gas limit allowed per transaction. This could result in the storage mapper becoming unusable, thus making all the functions that use this mapper unusable.

**! Possible fix to research**

```
As a general guideline, smart contract storage should not be used to
save large historical data; off-chain solutions are preferable for this
purpose.
```

**! Response**

```
Fixed.
```

**! Status**

```
Accepted & Closed.
```