verichains

SECURITY AUDIT OF

NFT AND MARKETPLACE SMARTCONTRACTS

200Lab
Education

Public Report

Feb 14, 2022

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 Feb 14, 2022. We would like to thank the 200Lab 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 NFT and Marketplace Smartcontracts. 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 one vulnerable issue in the contract code. 200Lab team has aknowledged and fixed the issue.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About NFT and Marketplace smartcontracts

200Lab Education is a software trainning institution. NFT and Marketplace smartcontracts is the smartcontracts belong to the blockchain course of 200Lab Education.

1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the ERC721 token and NFT Marketplace. It was conducted on the source code provided by the 200Lab team.

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

SHA256 Sum	File
8cfb1cc2b01cb817a8dcb8f329919b5e9472c90e7c69f041dd4e31527cd597d2	**MyNFT.sol**
d61806c76ba01c7f17b53806514910d20ed29282603d7db138531fd0e89d1d94	**Marketplace.sol**

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 NFT and Marketplace smartcontracts was written in Solidity language which requires version to be ^0.8.8.

2.1.1. Marketplace contract

This is the Marketplace in the NFT and Marketplace smartcontracts, which extends Ownable contract. With Ownable, by default, contract Owner is contract deployer, but he can transfer ownership to another address at any time.

The Marketplace allow users can list allowed NFT for selling by addOrder function and anybody can buy it with the listing price by executeOrder function. The owner of order can cancel the order by cancel function if no transaction has been executed. In additional, the order transaction will be charged fee when executed successfully and the fee will be transfered to feeRecipient address.

The owner of the contract can update the fee rate and fee recipient by updateFeeRate and updateFeeRecipient functions. He also can add payment token by calling addPaymentToken function.

2.1.2. MyNFT contract

This is the NFT contract in the NFT and Marketplace smartcontracts, which extends ERC721 and Ownable contract. With Ownable, by default, Token Owner is contract deployer, but he can transfer ownership to another address at any time.

The owner can mint new NFT and change the base URL by using mint and updateBaseTokenURI function.

2.2. Findings

During the audit process, the audit team found one vulnerability in the given version of the NFT and Marketplace smartcontracts.

2.2.1. Marketplace.sol - Missing checking order has been excecuted in cancel function CRITICAL

In the cancel function, there is no checking for the order has been sold or not so the owner of the order can call cancel function to receive NFT in case the NFT has been listed again by other owner of NFT.

```
function cancelOrder(uint256 orderId_) external {
    Order storage _order = orders[orderId_];
    require(_order.seller == _msgSender(), "NFTMarketplace: must be owner…
  ");
    uint256 _tokenId = _order.tokenId;
    delete orders[orderId_];
    nftContract.transferFrom(address(this), _msgSender(), _tokenId);
    emit OrderCancelled(orderId_);
}
```

RECOMMENDATION

Adding the checking whether the order has been sold or not.

```
function cancelOrder(uint256 orderId_) external {
    Order storage _order = orders[orderId_];
    require(_order.seller == _msgSender(), "NFTMarketplace: must be owner…
  ");
    // Add check if only buyer is zero or not
    require(_order.buyer == address(0), "NFTMarketplace: buyer must be ze…
  ro");
    uint256 _tokenId = _order.tokenId;
    delete orders[orderId_];
    nftContract.transferFrom(address(this), _msgSender(), _tokenId);
    emit OrderCancelled(orderId_);
}
```

UPDATES

- This issue has been acknowledged and fixed by the 200Lab team.

APPENDIX

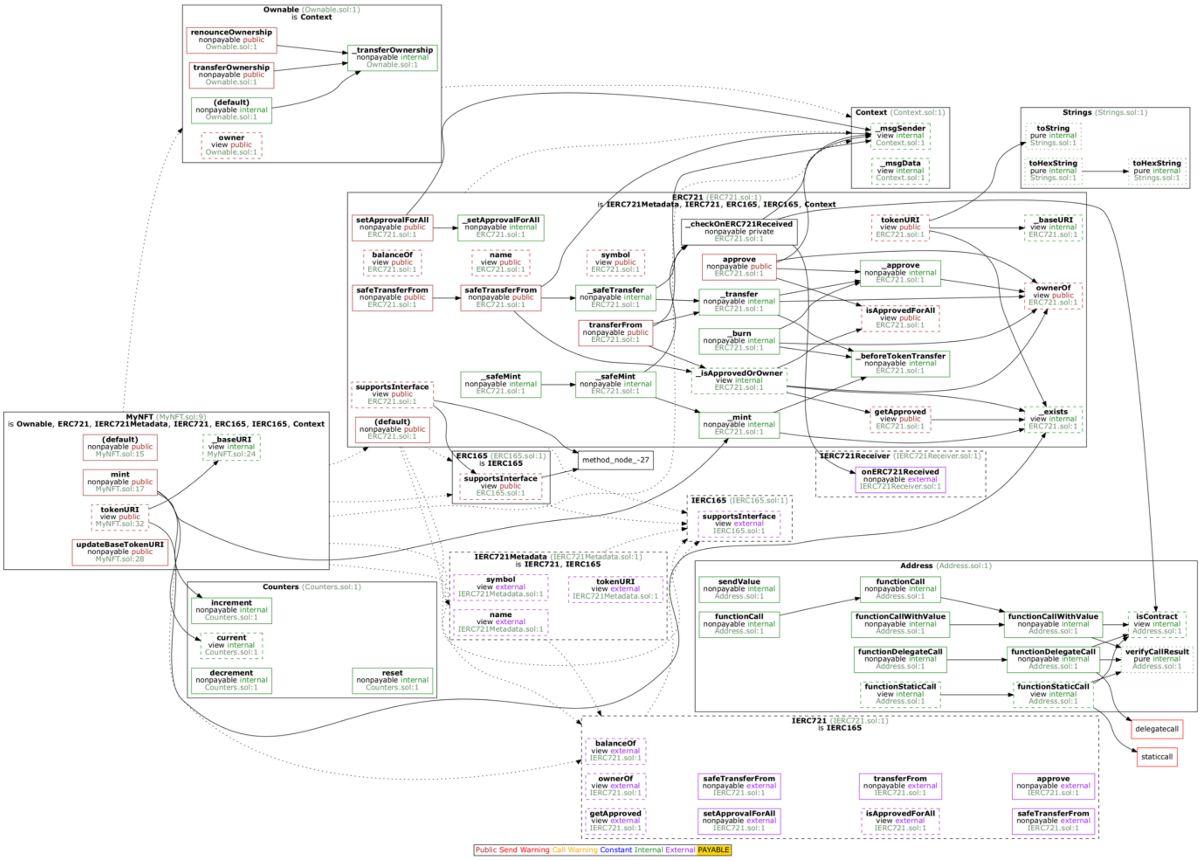
Ownable (Ownable.sol:1) is Context
renounceOwnership nonpayable public Ownable.sol:1
transferOwnership nonpayable public Ownable.sol:1
(default) nonpayable internal Ownable.sol:1
owner view public Ownable.sol:1
_transferOwnership nonpayable internal Ownable.sol:1
Context (Context.sol:1)
_msgSender view internal Context.sol:1
_msgData view internal Context.sol:1
Strings (Strings.sol:1)
toString pure internal Strings.sol:1
toHexString pure internal Strings.sol:1
toHexString pure internal Strings.sol:1
ERC721 (ERC721.sol:1) is IERC721Metadata, IERC721, ERC165, IERC165, Context
setApprovalForAll nonpayable public ERC721.sol:1
_setApprovalForAll nonpayable internal ERC721.sol:1
balanceOf view public ERC721.sol:1
name view public ERC721.sol:1
symbol view public ERC721.sol:1
safeTransferFrom nonpayable public ERC721.sol:1
safeTransferFrom nonpayable public ERC721.sol:1
_safeTransfer nonpayable internal ERC721.sol:1
transferFrom nonpayable public ERC721.sol:1
_checkOnERC721Received nonpayable private ERC721.sol:1
approve nonpayable public ERC721.sol:1
_transfer nonpayable internal ERC721.sol:1
_burn nonpayable internal ERC721.sol:1
_isApprovedOrOwner view internal ERC721.sol:1
_mint nonpayable internal ERC721.sol:1
tokenURI view public ERC721.sol:1
_baseURI view internal ERC721.sol:1
_approve nonpayable internal ERC721.sol:1
ownerOf view public ERC721.sol:1
isApprovedForAll view public ERC721.sol:1
_beforeTokenTransfer nonpayable internal ERC721.sol:1
getApproved view public ERC721.sol:1
_exists view internal ERC721.sol:1
supportsInterface view public ERC721.sol:1
_safeMint nonpayable internal ERC721.sol:1
_safeMint nonpayable internal ERC721.sol:1
(default) nonpayable public ERC721.sol:1
method_node_-27
MyNFT (MyNFT.sol:9) is Ownable, ERC721, IERC721Metadata, IERC721, ERC165, IERC165, Context
(default) nonpayable public MyNFT.sol:15
_baseURI view internal MyNFT.sol:24
mint nonpayable public MyNFT.sol:17
tokenURI view public MyNFT.sol:32
updateBaseTokenURI nonpayable public MyNFT.sol:28
ERC165 (ERC165.sol:1) is IERC165
supportsInterface view public ERC165.sol:1
IERC721Receiver (IERC721Receiver.sol:1)
onERC721Received nonpayable external IERC721Receiver.sol:1
IERC165 (IERC165.sol:1)
supportsInterface view external IERC165.sol:1
IERC721Metadata (IERC721Metadata.sol:1) is IERC721, IERC165
symbol view external IERC721Metadata.sol:1
tokenURI view external IERC721Metadata.sol:1
name view external IERC721Metadata.sol:1
Counters (Counters.sol:1)
increment nonpayable internal Counters.sol:1
current view internal Counters.sol:1
decrement nonpayable internal Counters.sol:1
reset nonpayable internal Counters.sol:1
Address (Address.sol:1)
sendValue nonpayable internal Address.sol:1
functionCall nonpayable internal Address.sol:1
functionCall nonpayable internal Address.sol:1
functionCallWithValue nonpayable internal Address.sol:1
functionCallWithValue nonpayable internal Address.sol:1
isContract view internal Address.sol:1
functionDelegateCall nonpayable internal Address.sol:1
functionDelegateCall nonpayable internal Address.sol:1
verifyCallResult pure internal Address.sol:1
functionStaticCall view internal Address.sol:1
functionStaticCall view internal Address.sol:1
delegatecall
staticcall
IERC721 (IERC721.sol:1) is IERC165
balanceOf view external IERC721.sol:1
ownerOf view external IERC721.sol:1
getApproved view external IERC721.sol:1
safeTransferFrom nonpayable external IERC721.sol:1
setApprovalForAll nonpayable external IERC721.sol:1
transferFrom nonpayable external IERC721.sol:1
isApprovedForAll view external IERC721.sol:1
approve nonpayable external IERC721.sol:1
safeTransferFrom nonpayable external IERC721.sol:1
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 1. MyNFT graph

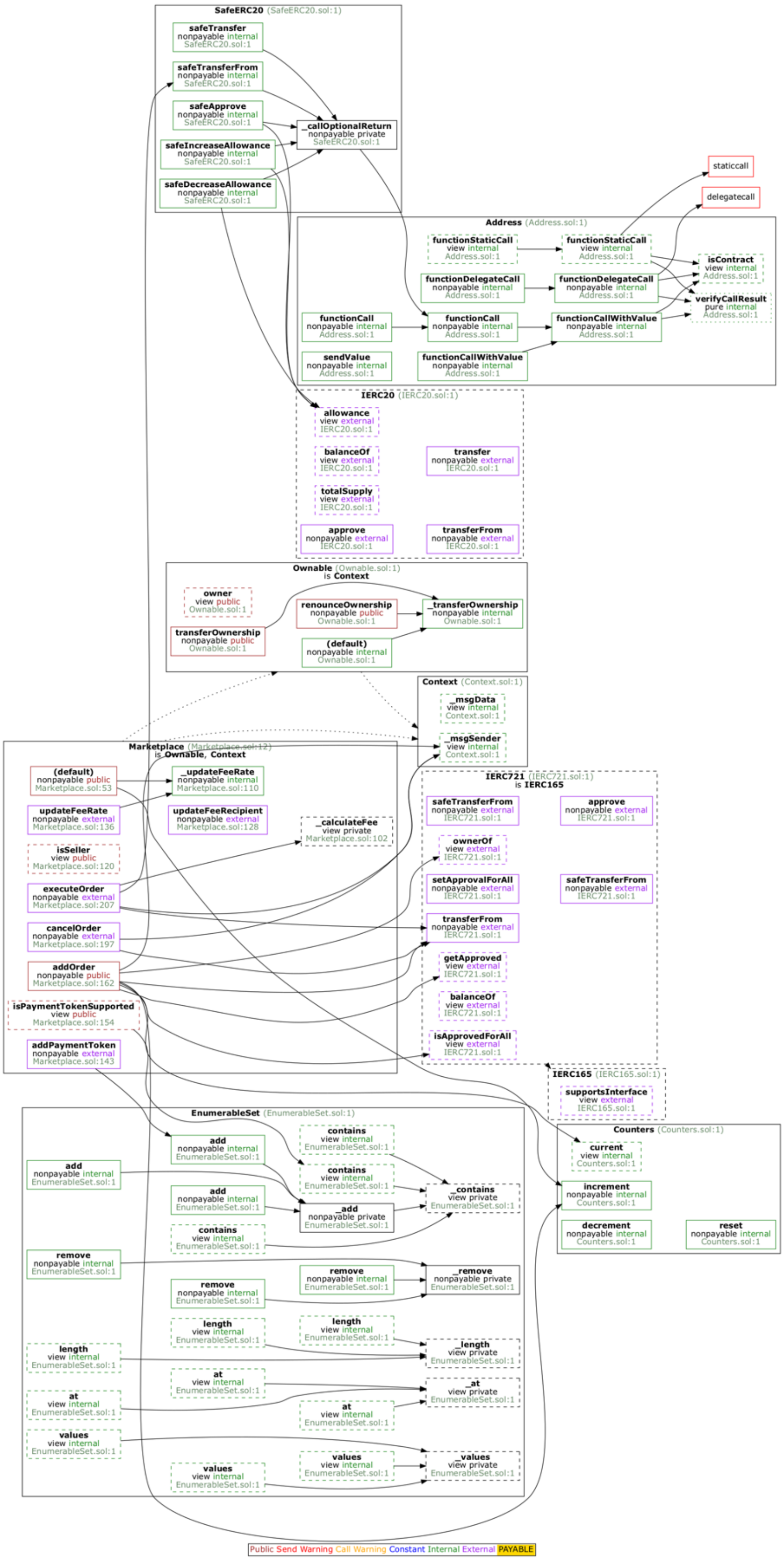
SafeERC20 (SafeERC20.sol:1)
safeTransfer nonpayable internal SafeERC20.sol:1
safeTransferFrom nonpayable internal SafeERC20.sol:1
safeApprove nonpayable internal SafeERC20.sol:1
safeIncreaseAllowance nonpayable internal SafeERC20.sol:1
safeDecreaseAllowance nonpayable internal SafeERC20.sol:1
_callOptionalReturn nonpayable private SafeERC20.sol:1
staticcall
delegatecall
Address (Address.sol:1)
functionStaticCall view internal Address.sol:1
functionStaticCall view internal Address.sol:1
isContract view internal Address.sol:1
functionDelegateCall nonpayable internal Address.sol:1
functionDelegateCall nonpayable internal Address.sol:1
verifyCallResult pure internal Address.sol:1
functionCall nonpayable internal Address.sol:1
functionCall nonpayable internal Address.sol:1
functionCallWithValue nonpayable internal Address.sol:1
sendValue nonpayable internal Address.sol:1
functionCallWithValue nonpayable internal Address.sol:1
IERC20 (IERC20.sol:1)
allowance view external IERC20.sol:1
balanceOf view external IERC20.sol:1
transfer nonpayable external IERC20.sol:1
totalSupply view external IERC20.sol:1
approve nonpayable external IERC20.sol:1
transferFrom nonpayable external IERC20.sol:1
Ownable (Ownable.sol:1) is Context
owner view public Ownable.sol:1
renounceOwnership nonpayable public Ownable.sol:1
_transferOwnership nonpayable internal Ownable.sol:1
transferOwnership nonpayable public Ownable.sol:1
(default) nonpayable internal Ownable.sol:1
Context (Context.sol:1)
_msgData view internal Context.sol:1
_msgSender view internal Context.sol:1
Marketplace (Marketplace.sol:12) is Ownable, Context
(default) nonpayable public Marketplace.sol:53
_updateFeeRate nonpayable internal Marketplace.sol:110
updateFeeRate nonpayable external Marketplace.sol:136
updateFeeRecipient nonpayable external Marketplace.sol:128
_calculateFee view private Marketplace.sol:102
isSeller view public Marketplace.sol:120
executeOrder nonpayable external Marketplace.sol:207
cancelOrder nonpayable external Marketplace.sol:197
addOrder nonpayable public Marketplace.sol:162
isPaymentTokenSupported view public Marketplace.sol:154
addPaymentToken nonpayable external Marketplace.sol:143
IERC721 (IERC721.sol:1) is IERC165
safeTransferFrom nonpayable external IERC721.sol:1
approve nonpayable external IERC721.sol:1
ownerOf view external IERC721.sol:1
setApprovalForAll nonpayable external IERC721.sol:1
safeTransferFrom nonpayable external IERC721.sol:1
transferFrom nonpayable external IERC721.sol:1
getApproved view external IERC721.sol:1
balanceOf view external IERC721.sol:1
isApprovedForAll view external IERC721.sol:1
IERC165 (IERC165.sol:1)
supportsInterface view external IERC165.sol:1
Counters (Counters.sol:1)
current view internal Counters.sol:1
increment nonpayable internal Counters.sol:1
decrement nonpayable internal Counters.sol:1
reset nonpayable internal Counters.sol:1
EnumerableSet (EnumerableSet.sol:1)
contains view internal EnumerableSet.sol:1
add nonpayable internal EnumerableSet.sol:1
add nonpayable internal EnumerableSet.sol:1
contains view internal EnumerableSet.sol:1
_contains view private EnumerableSet.sol:1
add nonpayable internal EnumerableSet.sol:1
_add nonpayable private EnumerableSet.sol:1
contains view internal EnumerableSet.sol:1
remove nonpayable internal EnumerableSet.sol:1
remove nonpayable internal EnumerableSet.sol:1
_remove nonpayable private EnumerableSet.sol:1
remove nonpayable internal EnumerableSet.sol:1
length view internal EnumerableSet.sol:1
length view internal EnumerableSet.sol:1
_length view private EnumerableSet.sol:1
length view internal EnumerableSet.sol:1
at view internal EnumerableSet.sol:1
_at view private EnumerableSet.sol:1
at view internal EnumerableSet.sol:1
at view internal EnumerableSet.sol:1
values view internal EnumerableSet.sol:1
values view internal EnumerableSet.sol:1
_values view private EnumerableSet.sol:1
values view internal EnumerableSet.sol:1
Public Send Warning Call Warning Constant Internal External PAYABLE

Image 2. Marketplace graph

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Feb 14, 2022*	Public Report	Verichains Lab

Table 2. Report versions history