Smart Contract
Security Audit
V1

The Bookmakers NFT Smart Contract

2/4/2022

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

Table of Contents

Table of Contents

Background

Project Information

- NFT Information
- Executive Summary

File and Function Level Report

- **File in Scope:**

Issues Checking Status

- Severity Definitions
- Audit Findings

Automatic testing

- Testing proves
- Inheritance graph
- Call graph

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Project Information

- **Platform**: Ethereum
- **Contract Address**: 0x2eF9044b00EE78FA38066b07089Af6419bc167cA
- **Code:**

https://github.com/Saferico/Smart-Contracts-for-Projects/blob/main/BMC.sol

NFT Information

- Name: BMC
- Total Supply: 7777
- Holders:
- Total transactions:

Contracts address deployed to test net (ETH)

The Bookmarkers NFT Smart contract on ETH test net to test write functions by the auditor.

https://rinkeby.etherscan.io/address/0x2ef9044b00ee78fa38066b07089af6419bc167ca

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Well-Secured**.

Well Secured	✓
Secured	
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 2 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

BMC.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
BMC.sol	**b71f48df51009c02a9415e22dace141302a809b982f48fe4975edb0eefbabcd2**	0x2eF9044b00EE78FA38066b07089Af6419bc167cA

- Contract: BMC
- Inherit:ERC721, ERC721Enumerable, Ownable
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
name	✓	Read / public	**Passed**
symbol	✓	Read / public	**Passed**
addressMintedBalance	✓	Read / public	**Passed**
supportsInterface	✓	Read / public	**Passed**
CURR_MINT_COST_1	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
CURR_MINT_COST_2	✓	Read / public	**Passed**
getInformations	✓	Read / public	**Passed**
getApprovedForAll	✓	Read / public	**Passed**
ownerOf	✓	Read / public	**Passed**
getApproved	✓	Read / public	**Passed**

tokenURI	✓	Read / public	**Passed**
tokenByIndex	✓	Read / public	**Passed**
tokenOfOwnerByIndex	✓	Read / public	**Passed**
hasSaleStarted	✓	Read / public	**Passed**
MAX_TOKENS_VIP	✓	Read / public	**Passed**
MAX_TOKENS	✓	Read / public	**Passed**
baseURI	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
verificationHash1	✓	Read / public	**Passed**
onlyWhitelisted	✓	Read / public	**Passed**
verificationHash2	✓	Read / public	**Passed**
walletOfOwner	✓	Read / public	**Passed**
reserveVIP	✓	Write / public	**Passed**
approve	✓	Write / public	**Passed**
safeTransferFrom	✓	Write / public	**Passed**
safeTransferFrom	✓	Write / public	**Passed**
setBaseURI	✓	Write / public	**Passed**
mintNFT1	✓	Write / payable	**Passed**
transferOwnership	✓	Write / public	**Passed**
setApprovalForAll	✓	Write / public	**Passed**
transferFrom	✓	Write / public	**Passed**
mintNFT2	✓	Write / payable	**Passed**
renounceOwnership	✓	Write / public	**Passed**
withdraw	✓	Write / public	**Passed**
setNewRound	✓	Write / public	**Passed**
setSaleStarted	✓	Write / public	**Passed**
Giveaway	✓	Write / public	**Passed**

setVerificationHash	✓	Write / public	**Passed**
setOnlyWhitelisted	✓	Write / public	**Passed**

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	**Passed**
2	Race conditions and Reentrancy. Cross-function race conditions.	**Passed**
3	Possible delays in data delivery.	**Passed**
4	Oracle calls.	**Passed**
5	Design Logic.	**Passed**
6	Timestamp dependence.	**Passed with Notes**
7	Integer Overflow and Underflow.	**Passed**
8	DoS with Revert.	**Passed**
9	DoS with block gas limit.	**Passed**
10	Methods execution permissions.	**Passed**
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	**Passed**
12	The impact of the exchange rate on the logic.	**Passed**
13	Private user data leaks.	**Passed**
14	Malicious Event log.	**Passed**
15	Scoping and Declarations.	**Passed**
16	Uninitialized storage pointers.	**Passed**
17	Arithmetic accuracy.	**Passed**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have 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 tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can’t have significant impact on execution
Note	Lowest-level vulnerabilities, code style violations and info statements can’t affect smart contract execution and can be ignored.

Audit Findings

Critical:

No critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found

Medium:

No Medium severity vulnerabilities were found

Low:

#Missing zero address validation

Description

When the owner wants to Giveaway and reserveVIP for the investors it has to check for the zero address to make, he didn't mint for the burn address. Otherwise, the mint function will act like the burn function.

```
function reserveVIP(uint numTokens, address recipient) public onlyOwner {
        require((currentVIPs + numTokens) <= MAX_TOKENS_VIP, "Exceeded VIP
supply");
        uint index;
        for(index = 1; index <= numTokens; index++) {
            _currentToken++;
            currentVIPs = currentVIPs + 1;
            uint theToken = currentVIPs + MAX_TOKENS;
            addressMintedBalance[recipient]++;
            _safeMint(recipient, theToken);
        }
    }

    function Giveaways(uint numTokens, address recipient) public onlyOwner {
        require((_currentToken + numTokens) <= MAX_TOKENS, "Exceeded supply");
        uint index;
        // Reserved for the people who helped build this project
        for(index = 1; index <= numTokens; index++) {
            _currentToken++;
            uint theToken = drawIndex();
            addressMintedBalance[recipient]++;
            _safeMint(recipient, theToken);
        }
    }
```

Remediation

Use the require statement to check for zero addresses.

```
require(recipient!= address(0), "Not Mint for the zero address");
```

Status: Closed. Fixed in version2.

#Use of block.timestamp for comparisons

Description
The value of block.timestamp can be manipulated by the miner.
And conditions with strict equality is difficult to achieve -
block.timestamp

Remediation
Avoid use of block.timestamp

Status: Acknowledged

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Unnecessary import of ERC721 library

Description
The main contract inherits: ERC721, and Ownable, and ERC721Enumerable which is already import ERC721 library, so no need to import it again in the main contract.

Remediation
Remove unnecessary library from the main contract save some gas fees.

Status: Closed. Fixed in version2.

Automatic Testing

1- Check for security

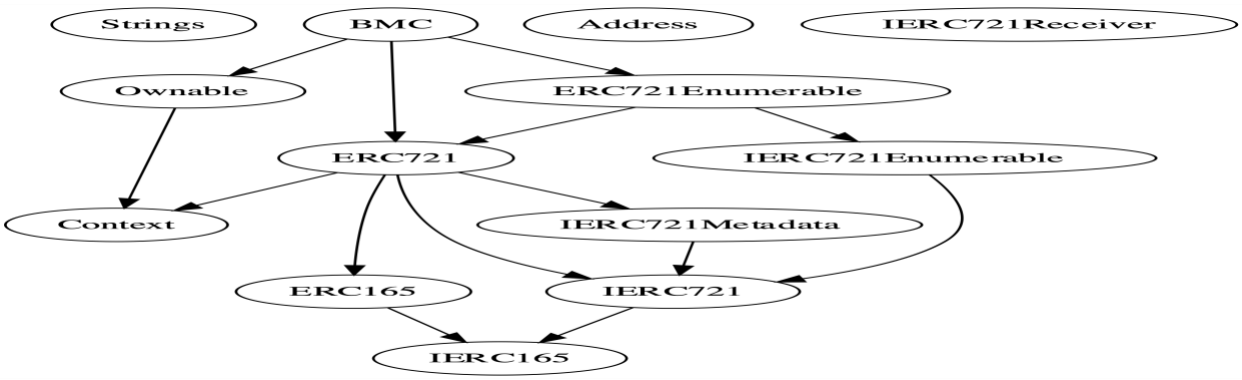
b71f48df51009c02a9415e22dace141302a809b982f48fe4975edb0eefbabcd2

File: BMC.sol | Language: solidity | Size: 7497 bytes | Date: 2022-04-02T08:39:39.100Z

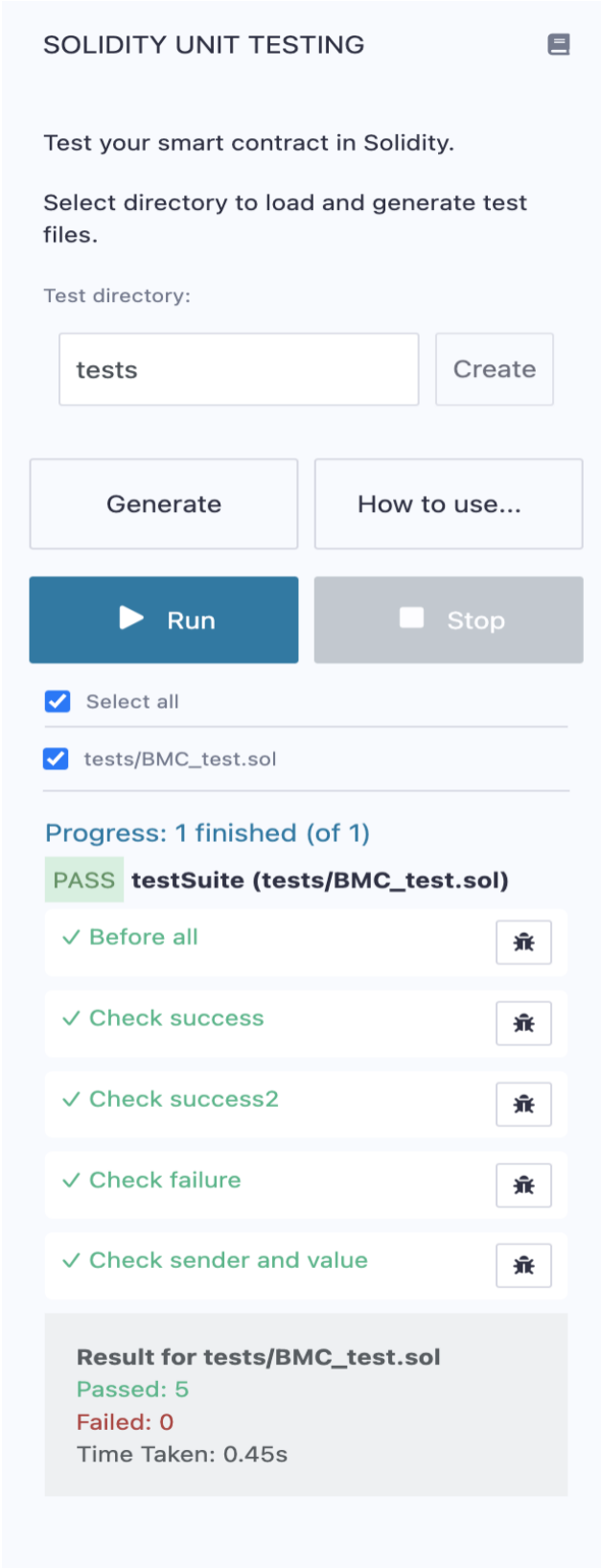
Critical	High	Medium	Low	Note
0	0	0	0	0

2- SOLIDITY STATIC ANALYSIS

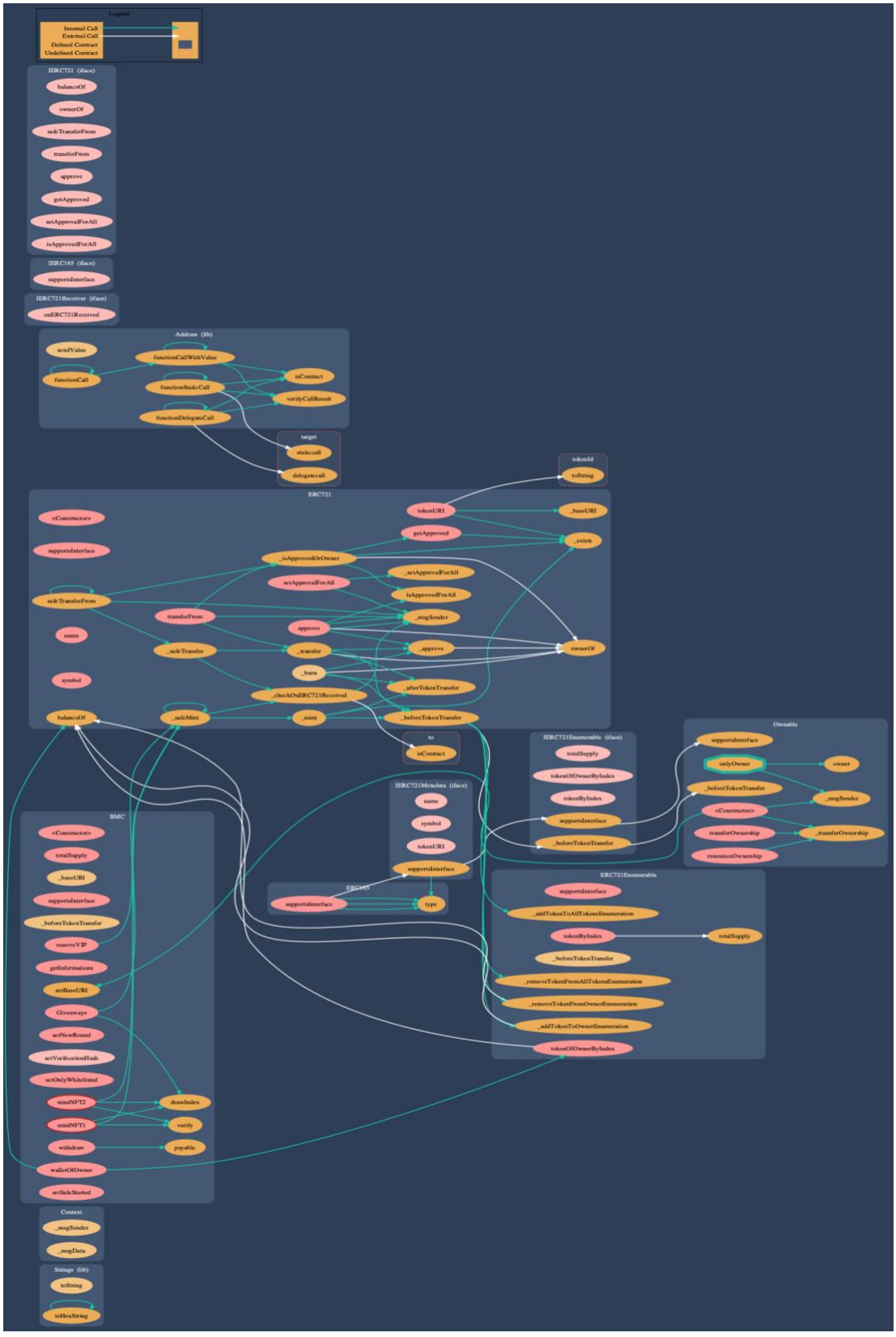
3- Inheritance graph



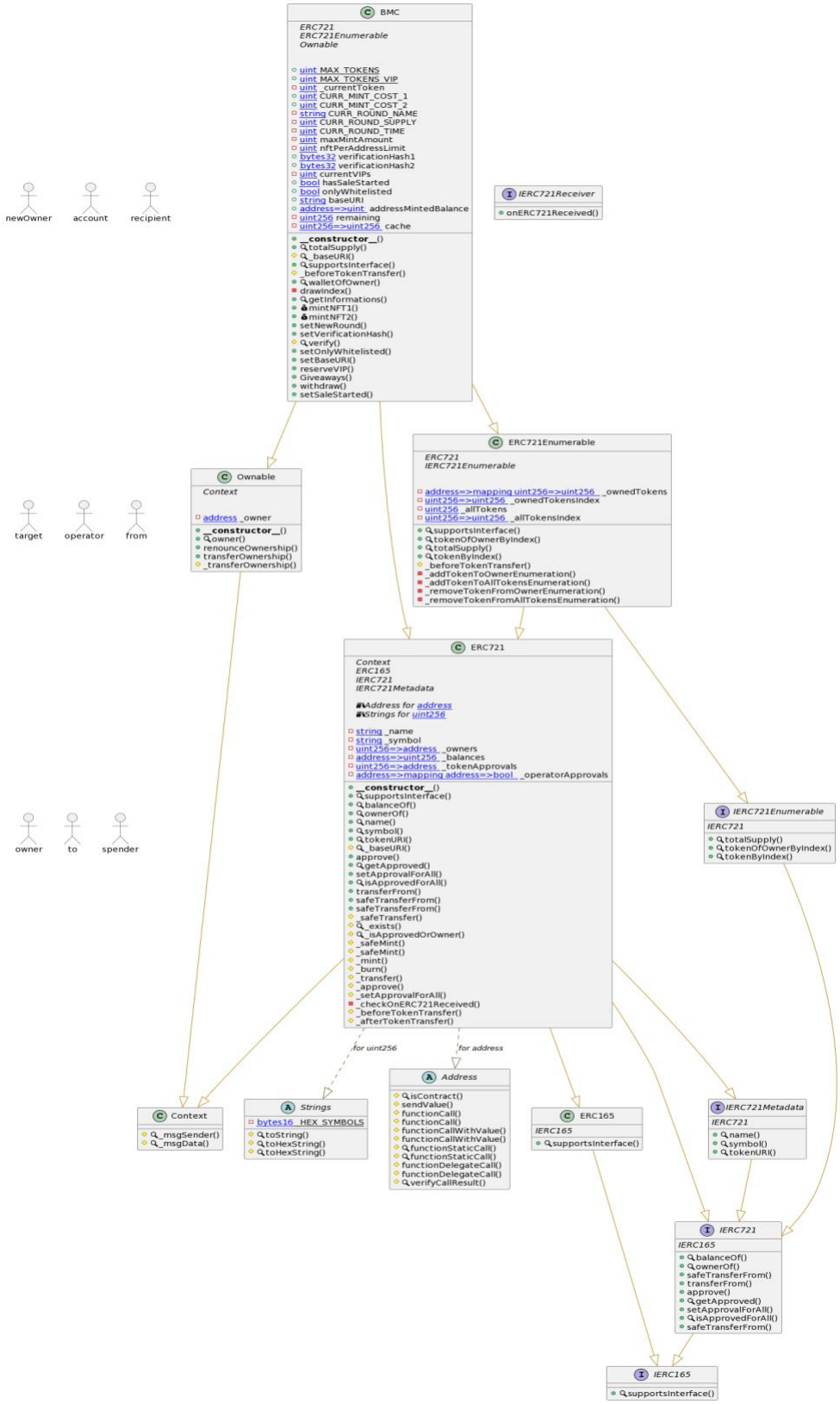
4- SOLIDITY UNIT TESTING



5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IERC721 (iface)
balanceOf
ownerOf
safeTransferFrom
transferFrom
approve
getApproved
setApprovalForAll
isApprovedForAll
IERC165 (iface)
supportsInterface
IERC721Receiver (iface)
onERC721Received
Address (lib)
sendValue
functionCallWithValue
functionCall
isContract
functionStaticCall
verifyCallResult
functionDelegateCall
target
staticcall
delegatecall
tokenId
toString
ERC721
<Constructor>
supportsInterface
safeTransferFrom
name
symbol
balanceOf
tokenURI
getApproved
_isApprovedOrOwner
setApprovalForAll
transferFrom
approve
_safeTransfer
_transfer
_burn
_checkOnERC721Received
_safeMint
_mint
_baseURI
_exists
_setApprovalForAll
isApprovedForAll
_msgSender
_approve
ownerOf
_afterTokenTransfer
_beforeTokenTransfer
to
isContract
IERC721Enumerable (iface)
totalSupply
tokenOfOwnerByIndex
tokenByIndex
supportsInterface
_beforeTokenTransfer
Ownable
supportsInterface
onlyOwner
owner
_beforeTokenTransfer
_msgSender
<Constructor>
transferOwnership
_transferOwnership
renounceOwnership
IERC721Metadata (iface)
name
symbol
tokenURI
supportsInterface
ERC165
supportsInterface
type
ERC721Enumerable
supportsInterface
_addTokenToAllTokensEnumeration
tokenByIndex
totalSupply
_beforeTokenTransfer
_removeTokenFromAllTokenEnumeration
_removeTokenFromOwnerEnumeration
_addTokenToOwnerEnumeration
tokenOfOwnerByIndex
BMC
<Constructor>
totalSupply
_baseURI
supportsInterface
_beforeTokenTransfer
reserveVIP
getInformations
setBaseURI
Giveaways
setNewRound
setVerificationHash
setOnlyWhitelisted
mintNFT2
mintNFT1
withdraw
walletOfOwner
setSaleStarted
drawIndex
verify
payable
Context
_msgSender
_msgData
Strings (lib)
toString
toHexString

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
16279055  =>  isContract(address)
6900a3ae  =>  toString(uint256)
8fba8d5c  =>  toHexString(uint256)
63e1cbea  =>  toHexString(uint256,uint256)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
8da5cb5b  =>  owner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
d29d44ee  =>  _transferOwnership(address)
24a084df  =>  sendValue(address,uint256)
a0b5ffb0  =>  functionCall(address,bytes)
241b5886  =>  functionCall(address,bytes,string)
2a011594  =>  functionCallWithValue(address,bytes,uint256)
d525ab8a  =>  functionCallWithValue(address,bytes,uint256,string)
c21d36f3  =>  functionStaticCall(address,bytes)
dbc40fb9  =>  functionStaticCall(address,bytes,string)
ee33b7e2  =>  functionDelegateCall(address,bytes)
57387df0  =>  functionDelegateCall(address,bytes,string)
946b5793  =>  verifyCallResult(bool,bytes,string)
150b7a02  =>  onERC721Received(address,address,uint256,bytes)
01ffc9a7  =>  supportsInterface(bytes4)
70a08231  =>  balanceOf(address)
6352211e  =>  ownerOf(uint256)
42842e0e  =>  safeTransferFrom(address,address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
095ea7b3  =>  approve(address,uint256)
081812fc  =>  getApproved(uint256)
a22cb465  =>  setApprovalForAll(address,bool)
e985e9c5  =>  isApprovedForAll(address,address)
b88d4fde  =>  safeTransferFrom(address,address,uint256,bytes)
18160ddd  =>  totalSupply()
2f745c59  =>  tokenOfOwnerByIndex(address,uint256)
4f6ccce7  =>  tokenByIndex(uint256)
06fdde03  =>  name()
95d89b41  =>  symbol()
c87b56dd  =>  tokenURI(uint256)
743976a0  =>  _baseURI()
24b6b8c0  =>  _safeTransfer(address,address,uint256,bytes)
f8e76cc0  =>  _exists(uint256)
4cdc9549  =>  _isApprovedOrOwner(address,uint256)
b3e1c718  =>  _safeMint(address,uint256)
6a4f832b  =>  _safeMint(address,uint256,bytes)
4e6ec247  =>  _mint(address,uint256)
9b1f9e74  =>  _burn(uint256)
30e0789e  =>  _transfer(address,address,uint256)
7b7d7225  =>  _approve(address,uint256)
8c4e3f32  =>  _setApprovalForAll(address,address,bool)
1fd01de1  =>  _checkOnERC721Received(address,address,uint256,bytes)
cad3be83  =>  _beforeTokenTransfer(address,address,uint256)
8f811a1c  =>  _afterTokenTransfer(address,address,uint256)
```

```
69025b5f  =>  _addTokenToOwnerEnumeration(address,uint256)
e03d890b  =>  _addTokenToAllTokensEnumeration(uint256)
68df0d53  =>  _removeTokenFromOwnerEnumeration(address,uint256)
4cbb4a0a  =>  _removeTokenFromAllTokensEnumeration(uint256)
438b6300  =>  walletOfOwner(address)
cbd53315  =>  drawIndex()
3c5e310b  =>  getInformations()
84291a1b  =>  mintNFT1(uint256,bytes32[])
3b3bd73b  =>  mintNFT2(uint256,bytes32[])
14a2309c  =>
setNewRound(uint256,uint256,uint256,string,uint256,uint256,uint256,bool,bool)
05c44b1f  =>  setVerificationHash(bytes32,bytes32)
0a02831c  =>  verify(bytes32,bytes32[],bytes32)
3c952764  =>  setOnlyWhitelisted(bool)
55f804b3  =>  setBaseURI(string)
bff84fc5  =>  reserveVIP(uint256,address)
cc0b8d15  =>  Giveaways(uint256,address)
2e1a7d4d  =>  withdraw(uint256)
a854ffba  =>  setSaleStarted(bool)
```

Automatic general report

```
Files Description Table
|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/BMC.sol |
c71397688b1135b7c98674e52294bbfd859d1830 |
 Contracts Description Table
|  Contract  |         Type        |       Bases      |                  |
|
|:----------:|:-------------------:|:----------------:|:----------------:|:--------
-------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |
**Modifiers**  |
||||||
| **Strings** | Library |  |||
| └ | toString | Internal 🔒 |   | |
| └ | toHexString | Internal 🔒 |   | |
| └ | toHexString | Internal 🔒 |   | |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | Internal 🔒 |   | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑  |NO❗ |
| └ | owner | Public ❗ |   |NO❗ |
| └ | renounceOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | _transferOwnership | Internal 🔒 | 🛑  | |
||||||
| **Address** | Library |  |||
| └ | isContract | Internal 🔒 |   | |
| └ | sendValue | Internal 🔒 | 🛑  | |
| └ | functionCall | Internal 🔒 | 🛑  | |
| └ | functionCall | Internal 🔒 | 🛑  | |
| └ | functionCallWithValue | Internal 🔒 | 🛑  | |
| └ | functionCallWithValue | Internal 🔒 | 🛑  | |
| └ | functionStaticCall | Internal 🔒 |   | |
| └ | functionStaticCall | Internal 🔒 |   | |
| └ | functionDelegateCall | Internal 🔒 | 🛑  | |
| └ | functionDelegateCall | Internal 🔒 | 🛑  | |
| └ | verifyCallResult | Internal 🔒 |   | |
||||||
| **IERC721Receiver** | Interface |  |||
| └ | onERC721Received | External ❗ | 🛑  |NO❗ |
||||||
| **IERC165** | Interface |  |||
| └ | supportsInterface | External ❗ |   |NO❗ |
||||||
| **ERC165** | Implementation | IERC165 |||
| └ | supportsInterface | Public ❗ |   |NO❗ |
||||||
| **IERC721** | Interface | IERC165 |||
```


└	balanceOf	External ❗		NO❗
└	ownerOf	External ❗		NO❗
└	safeTransferFrom	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	approve	External ❗	🛑	NO❗
└	getApproved	External ❗		NO❗
└	setApprovalForAll	External ❗	🛑	NO❗
└	isApprovedForAll	External ❗		NO❗
└	safeTransferFrom	External ❗	🛑	NO❗
IERC721Enumerable	Interface	IERC721		
└	totalSupply	External ❗		NO❗
└	tokenOfOwnerByIndex	External ❗		NO❗
└	tokenByIndex	External ❗		NO❗
IERC721Metadata	Interface	IERC721		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	tokenURI	External ❗		NO❗
ERC721	Implementation	Context, ERC165, IERC721, IERC721Metadata		
└	<Constructor>	Public ❗	🛑	NO❗
└	supportsInterface	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	ownerOf	Public ❗		NO❗
└	name	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	tokenURI	Public ❗		NO❗
└	_baseURI	Internal 🔒		
└	approve	Public ❗	🛑	NO❗
└	getApproved	Public ❗		NO❗
└	setApprovalForAll	Public ❗	🛑	NO❗
└	isApprovedForAll	Public ❗		NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	safeTransferFrom	Public ❗	🛑	NO❗
└	safeTransferFrom	Public ❗	🛑	NO❗
└	_safeTransfer	Internal 🔒	🛑	
└	_exists	Internal 🔒		
└	_isApprovedOrOwner	Internal 🔒		
└	_safeMint	Internal 🔒	🛑	
└	_safeMint	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_transfer	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_setApprovalForAll	Internal 🔒	🛑	
└	_checkOnERC721Received	Private 🔐	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
ERC721Enumerable	Implementation	ERC721, IERC721Enumerable		
└	supportsInterface	Public ❗		NO❗
└	tokenOfOwnerByIndex	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	tokenByIndex	Public ❗		NO❗

└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_addTokenToOwnerEnumeration	Private 🔐	🛑	
└	_addTokenToAllTokensEnumeration	Private 🔐	🛑	
└	_removeTokenFromOwnerEnumeration	Private 🔐	🛑	
└	_removeTokenFromAllTokensEnumeration	Private 🔐	🛑	
BMC	Implementation	ERC721, ERC721Enumerable, Ownable		
└	<Constructor>	Public ❗	🛑	ERC721
└	totalSupply	Public ❗		NO❗
└	_baseURI	Internal 🔒		
└	supportsInterface	Public ❗		NO❗
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	walletOfOwner	Public ❗		NO❗
└	drawIndex	Private 🔐	🛑	
└	getInformations	Public ❗		NO❗
└	mintNFT1	Public ❗	💸	NO❗
└	mintNFT2	Public ❗	💸	NO❗
└	setNewRound	Public ❗	🛑	onlyOwner
└	setVerificationHash	External ❗	🛑	onlyOwner
└	verify	Internal 🔒		
└	setOnlyWhitelisted	Public ❗	🛑	onlyOwner
└	setBaseURI	Public ❗	🛑	onlyOwner
└	reserveVIP	Public ❗	🛑	onlyOwner
└	Giveaways	Public ❗	🛑	onlyOwner
└	withdraw	Public ❗	🛑	onlyOwner
└	setSaleStarted	Public ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💸	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “Well-secured”.

✓ No volatile code.
✓ Not many high severity issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Saferico hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.