Smart Contract Security Audit V1

BullHeaded NFTs Smart Contract Audit

29/3/2023

https://saferico.com/

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

–

Table of Contents

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**: 0xD4bD97AdC43C1393Bdda0eB55EaEFD5ca4A56c05
- **Code:**

https://goerli.etherscan.io/address/0xD4bD97AdC43C1393Bdda0eB55EaEFD5ca4A56c05#code

NFT Information

- Name: BullHeaded NFTs (BHEAD)
- MAX Supply: 2899
- Holders:
- Total transactions:

Contracts address deployed to test net (Ethereum)

BullHeaded NFTs Pass smart contract on Ethereum test net to test every function by the auditor.

https://goerli.etherscan.io/address/0x692373584402636bc27968c460c4216dd2e9fd96

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, 3 low, 0 very low-level issues and *1 note in all solidity files of the contract

The files:

BHEAD.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
BHEAD.sol	**a6d0021396313d37dfbf61d11a4bd6881aa85d666383993b30deef4ce47453ea**	0x692373584402636bc27968c460c4216dd2e9fd96

- Contract: BHEAD
- Inherit: ERC721A, 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**
getOwnershipData	✓	Read / public	**Passed**
supportsInterface	✓	Read / public	**Passed**
pauseMint	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
mintLimitPerWallet	✓	Read / public	**Passed**
tokenByIndex	✓	Read / public	**Passed**
getApprovedForAll	✓	Read / public	**Passed**
MAX_NFTS_SUPPLY	✓	Read / public	**Passed**
getApproved	✓	Read / public	**Passed**
ownerOf	✓	Read / public	**Passed**

tokenURI	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
ownerEarning	✓	Read / public	**Passed**
mintPrice	✓	Read / public	**Passed**
referrerEarning	✓	Read / public	**Passed**
unrevealURI	✓	Read / public	**Passed**
reveal	✓	Read / public	**Passed**
tokenOfOwnerByIndex	✓	Read / public	**Passed**
mintNFT	✓	Write / payable	**Passed**
approve	✓	Write / public	**Passed**
safeTransferFrom	✓	Write / public	**Passed**
safeTransferFrom	✓	Write / public	**Passed**
claimRefEarning	✓	Write / public	**Passed**
withdraw	✓	Write / public	**Passed**
setMintLimit	✓	Write / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
setApprovalForAll	✓	Write / public	**Passed**
transferFrom	✓	Write / public	**Passed**
setMintPrice	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
mintNFTForOwner	✓	Write / public	**Passed**
pause	✓	Write / public	**Passed**
unpause	✓	Write / public	**Passed**
Reveal	✓	Write / public	**Passed**
setBaseURI	✓	Write / public	**Passed**
unReveal	✓	Write / public	**Passed**
setUnrevealURI	✓	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 with Notes**
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:

#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

#Pragam version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.19 instead of ^0.8.0). contracts should be deployed with the same compiler version and flags that they have been tested the most with. Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, the latest compiler which may have higher risks of undiscovered bugs. Contracts may also be deployed by others and the pragma indicates the compiler version intended by the original authors.

Remediation

Remove the ^ sign to lock the pragma version.

Status: Acknowledged.

#Owner privileges (In the period when the owner isn't renounced)

Description

The owner can pause and un pause the contract.
The owner can change the price at any stage.

```
function setMintPrice(uint256 _newPrice) external onlyOwner() {
        mintPrice = _newPrice;
    }
    function pause() public onlyOwner {
        pauseMint = true;
    }
    function unPause() public onlyOwner {
        pauseMint = false;
    }
```

Remediation

Make these functions internal in next version or the team should announce the investors before doing anything to give them time if they want to do anything.

P.S: This issue is common to the majority of NFT smart contracts.

Status: Acknowledged.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Unnecessary import of Strings library

Description

The main contract inherits: Ownable, ERC721A which is already import Strings 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: Acknowledged.

Automatic Testing

1- Check for security

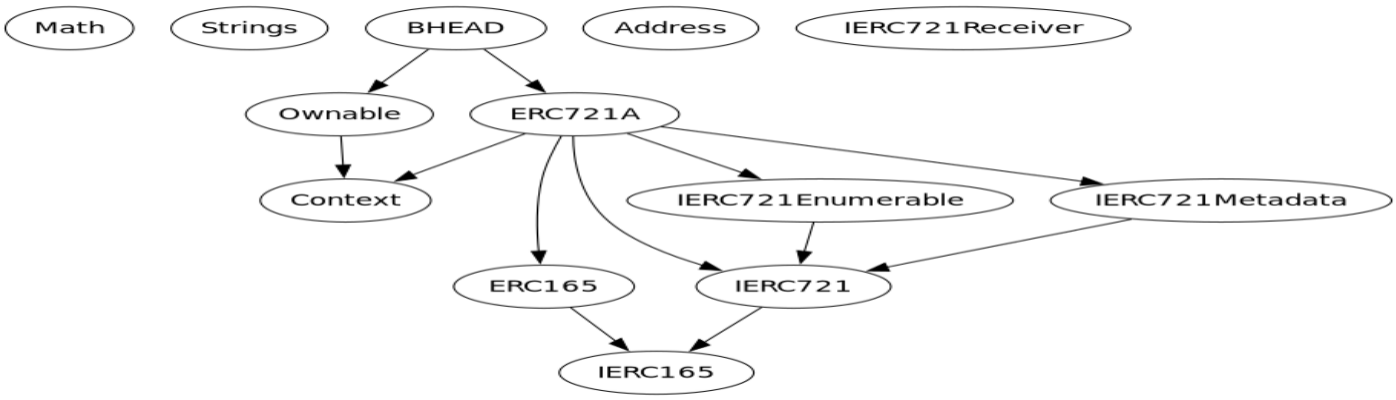
a6d0021396313d37dfbf61d11a4bd6881aa85d666383993b30deef4ce47453ea

File: BHEAD.sol | Language: solidity | Size: 3773 bytes | Date: 2023-03-29T12:12:21.197Z

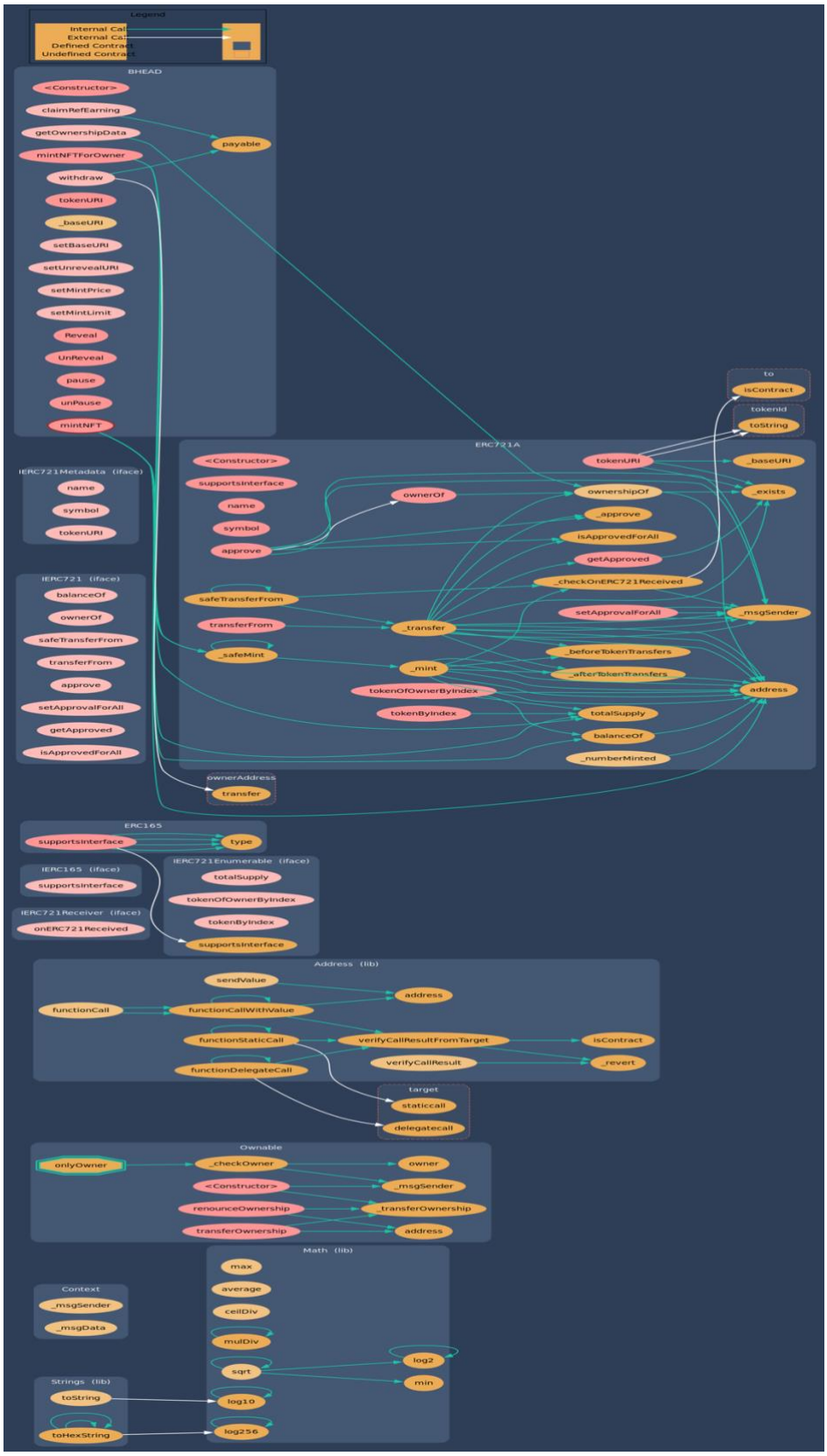
Critical	High	Medium	Low	Note
0	0	0	0	0

2- SOLIDITY STATIC ANALYSIS

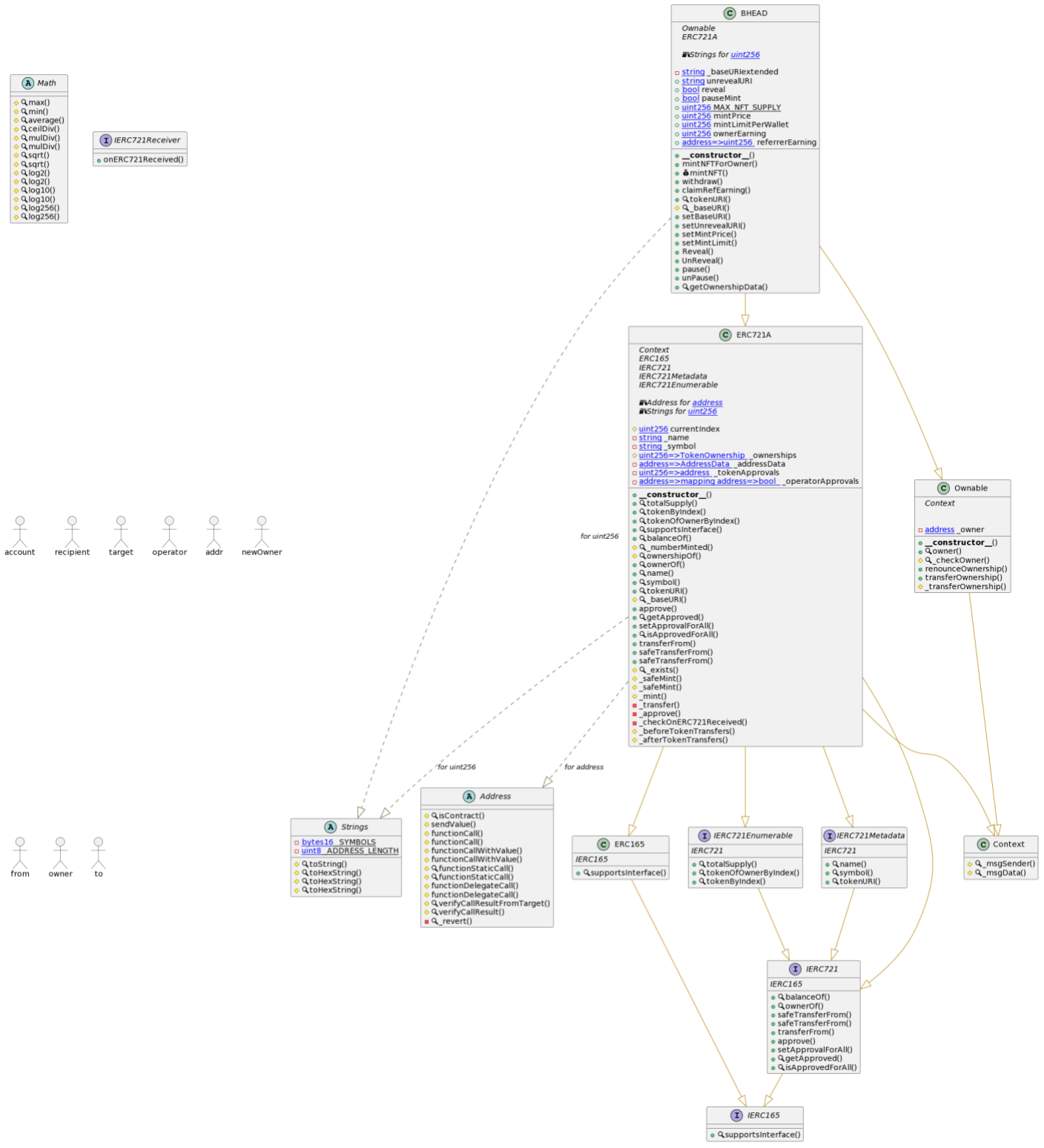
3- Inheritance graph



4- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
BHEAD
<Constructor>
claimRefEarning
getOwnershipData
mintNFTForOwner
withdraw
tokenURI
_baseURI
setBaseURI
setUnrevealURI
setMintPrice
setMintLimit
Reveal
UnReveal
pause
unPause
mintNFT
payable
to
isContract
tokenId
toString
ERC721A
<Constructor>
supportsInterface
name
symbol
approve
ownerOf
tokenURI
ownershipOf
_approve
isApprovedForAll
getApproved
_checkOnERC721Received
_baseURI
_exists
_msgSender
setApprovalForAll
safeTransferFrom
transferFrom
_transfer
_safeMint
_mint
_beforeTokenTransfers
_afterTokenTransfers
tokenOfOwnerByIndex
tokenByIndex
totalSupply
balanceOf
_numberMinted
address
IERC721Metadata (iface)
name
symbol
tokenURI
IERC721 (iface)
balanceOf
ownerOf
safeTransferFrom
transferFrom
approve
setApprovalForAll
getApproved
isApprovedForAll
ownerAddress
transfer
ERC165
supportsInterface
type
IERC165 (iface)
supportsInterface
IERC721Enumerable (iface)
totalSupply
tokenOfOwnerByIndex
tokenByIndex
supportsInterface
IERC721Receiver (iface)
onERC721Received
Address (lib)
sendValue
functionCall
functionCallWithValue
address
functionStaticCall
verifyCallResultFromTarget
isContract
verifyCallResult
_revert
functionDelegateCall
target
staticcall
delegatecall
Ownable
onlyOwner
_checkOwner
owner
<Constructor>
_msgSender
renounceOwnership
_transferOwnership
transferOwnership
address
Math (lib)
max
average
ceilDiv
mulDiv
sqrt
log2
min
log10
log256
Context
_msgSender
_msgData
Strings (lib)
toString
toHexString

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
16279055  =>  isContract(address)
6d5433e6  =>  max(uint256,uint256)
7ae2b5c7  =>  min(uint256,uint256)
2b7423ab  =>  average(uint256,uint256)
9cb35327  =>  ceilDiv(uint256,uint256)
aa9a0912  =>  mulDiv(uint256,uint256,uint256)
1db78456  =>  mulDiv(uint256,uint256,uint256,Rounding)
677342ce  =>  sqrt(uint256)
a902bc5e  =>  sqrt(uint256,Rounding)
5456bf13  =>  log2(uint256)
2ee6af53  =>  log2(uint256,Rounding)
ebdae5f9  =>  log10(uint256)
f86799ff  =>  log10(uint256,Rounding)
36cb4c48  =>  log256(uint256)
2910b3a1  =>  log256(uint256,Rounding)
6900a3ae  =>  toString(uint256)
8fba8d5c  =>  toHexString(uint256)
63e1cbea  =>  toHexString(uint256,uint256)
1bb0c665  =>  toHexString(address)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
8da5cb5b  =>  owner()
53a72975  =>  _checkOwner()
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)
1daa78c1  =>  verifyCallResultFromTarget(address,bool,bytes,string)
946b5793  =>  verifyCallResult(bool,bytes,string)
6cadf5e1  =>  _revert(bytes,string)
150b7a02  =>  onERC721Received(address,address,uint256,bytes)
01ffc9a7  =>  supportsInterface(bytes4)
70a08231  =>  balanceOf(address)
6352211e  =>  ownerOf(uint256)
b88d4fde  =>  safeTransferFrom(address,address,uint256,bytes)
42842e0e  =>  safeTransferFrom(address,address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
095ea7b3  =>  approve(address,uint256)
a22cb465  =>  setApprovalForAll(address,bool)
081812fc  =>  getApproved(uint256)
e985e9c5  =>  isApprovedForAll(address,address)
```

```
18160ddd  =>  totalSupply()
2f745c59  =>  tokenOfOwnerByIndex(address,uint256)
4f6ccce7  =>  tokenByIndex(uint256)
06fdde03  =>  name()
95d89b41  =>  symbol()
c87b56dd  =>  tokenURI(uint256)
4d388a98  =>  _numberMinted(address)
140364a1  =>  ownershipOf(uint256)
743976a0  =>  _baseURI()
f8e76cc0  =>  _exists(uint256)
b3e1c718  =>  _safeMint(address,uint256)
6a4f832b  =>  _safeMint(address,uint256,bytes)
de0d9900  =>  _mint(address,uint256,bytes,bool)
30e0789e  =>  _transfer(address,address,uint256)
f272404d  =>  _approve(address,uint256,address)
1fd01de1  =>  _checkOnERC721Received(address,address,uint256,bytes)
ef435773  =>  _beforeTokenTransfers(address,address,uint256,uint256)
08c018f7  =>  _afterTokenTransfers(address,address,uint256,uint256)
ad236d4c  =>  mintNFTForOwner(uint256)
669f5a51  =>  mintNFT(uint256,address)
3ccfd60b  =>  withdraw()
320f96c3  =>  claimRefEarning(address)
55f804b3  =>  setBaseURI(string)
97bc411c  =>  setUnrevealURI(string)
f4a0a528  =>  setMintPrice(uint256)
9e6a1d7d  =>  setMintLimit(uint256)
66b9f0d2  =>  Reveal()
672cb658  =>  UnReveal()
8456cb59  =>  pause()
f7b188a5  =>  unPause()
9231ab2a  =>  getOwnershipData(uint256)
```

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/BHEAD.sol	7a00bbd4f53a24d1a25c3a2ebf58adb70a7b01d2

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Math	Library			
└	max	Internal 🔒		
└	min	Internal 🔒		
└	average	Internal 🔒		
└	ceilDiv	Internal 🔒		
└	mulDiv	Internal 🔒		
└	mulDiv	Internal 🔒		
└	sqrt	Internal 🔒		
└	sqrt	Internal 🔒		
└	log2	Internal 🔒		
└	log2	Internal 🔒		
└	log10	Internal 🔒		
└	log10	Internal 🔒		
└	log256	Internal 🔒		
└	log256	Internal 🔒		
Strings	Library			
└	toString	Internal 🔒		
└	toHexString	Internal 🔒		
└	toHexString	Internal 🔒		
└	toHexString	Internal 🔒		
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	_checkOwner	Internal 🔒		
└	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 🔒	🛑	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		
└	_revert	Private 🔐		
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❗
└	safeTransferFrom	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	approve	External ❗	🛑	NO❗
└	setApprovalForAll	External ❗	🛑	NO❗
└	getApproved	External ❗		NO❗
└	isApprovedForAll	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❗
ERC721A	Implementation	Context, ERC165, IERC721, IERC721Metadata, IERC721Enumerable		
└	<Constructor>	Public ❗	🛑	NO❗
└	totalSupply	Public ❗		NO❗
└	tokenByIndex	Public ❗		NO❗
└	tokenOfOwnerByIndex	Public ❗		NO❗
└	supportsInterface	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	_numberMinted	Internal 🔒		
└	ownershipOf	Internal 🔒		

└	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❗
└	_exists	Internal 🔒		
└	_safeMint	Internal 🔒	🛑	
└	_safeMint	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_transfer	Private 🔐	🛑	
└	_approve	Private 🔐	🛑	
└	_checkOnERC721Received	Private 🔐	🛑	
└	_beforeTokenTransfers	Internal 🔒	🛑	
└	_afterTokenTransfers	Internal 🔒	🛑	
BHEAD	Implementation	Ownable, ERC721A		
└	<Constructor>	Public ❗	🛑	ERC721A
└	mintNFTForOwner	Public ❗	🛑	onlyOwner
└	mintNFT	Public ❗	💰	NO❗
└	withdraw	External ❗	🛑	onlyOwner
└	claimRefEarning	External ❗	🛑	NO❗
└	tokenURI	Public ❗		NO❗
└	_baseURI	Internal 🔒		
└	setBaseURI	External ❗	🛑	onlyOwner
└	setUnrevealURI	External ❗	🛑	onlyOwner
└	setMintPrice	External ❗	🛑	onlyOwner
└	setMintLimit	External ❗	🛑	onlyOwner
└	Reveal	Public ❗	🛑	onlyOwner
└	UnReveal	Public ❗	🛑	onlyOwner
└	pause	Public ❗	🛑	onlyOwner
└	unPause	Public ❗	🛑	onlyOwner
└	getOwnershipData	External ❗		NO❗

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.

✓ No 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.