



AUDIT COMPANY



SMART CONTRACT SECURITY AUDIT

Celestials Stellar Club

Mach 2022

CheckDot



Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at 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 us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER: 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 CheckDot and its affiliates (including CheckDot verifiers, shareholders, employees, directors, officers and other representatives) owe no duty of care towards you or any other person, nor does CheckDot 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 CheckDot 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, CheckDot hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against CheckDot, 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.



Background

CheckDot was commissioned by MagicBirdsToken to perform an audit of smart contracts:

Mainnet Deployed Contract:

<https://etherscan.io/address/0x0858DFC6c88137755049155426881dc2B04F89fc>

Github:

<https://github.com/CelestialsStellarClub/CelestialsNFTContract/tree/main>

Commit Hash:

5860b725d5c73a2d5f43ae91f15a542cba48d0a2

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.
- Verify that the ERC721 structure is followed.

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.



Issues Checking Status

Issue description	Checking status
1. Compiler errors.	Passed
2. Race conditions and Reentrancy. Cross-function race conditions.	Passed
3. Possible delays in data delivery.	Passed
4. Oracle calls.	None
5. Front running.	None
6. Timestamp dependence.	Passed
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 of the contract.	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
18. Design Logic.	Passed
19. Cross-function race conditions.	Passed
20. Safe Open Zeppelin contracts implementation and usage.	Passed
21. Fallback function security.	Passed
22. Fees detection	Passed



SMART CONTRACT AUDIT

```
In CelestialsContract.walletOfOwner(address) var i (contracts/CelestialsContract.sol#1323) is a local variable never initialized

CelestialsContract.constructor(string,string,string,string).name (contracts/CelestialsContract.sol#1276) shadows:
- ERC721_name (contracts/CelestialsContract.sol#637) (state variable)
CelestialsContract.constructor(string,string,string,string).symbol (contracts/CelestialsContract.sol#1277) shadows:
- ERC721_symbol (contracts/CelestialsContract.sol#640) (state variable)
CelestialsContract.walletOfOwner(address).owner (contracts/CelestialsContract.sol#1316) shadows:
- Ownable_owner (contracts/CelestialsContract.sol#1207) (state variable)

ERC721_checkOnERC721Received(address,address,uint256,bytes) (contracts/CelestialsContract.sol#983-1004) has external calls inside a loop:
IERC721Receiver(to).onERC721Received(msgSender(),from,tokenId,data) (contracts/CelestialsContract.sol#990-1000)

Variable 'ERC721_checkOnERC721Received(address,address,uint256,bytes).retval (contracts/CelestialsContract.sol#990)' in
ERC721_checkOnERC721Received(address,address,uint256,bytes) (contracts/CelestialsContract.sol#983-1004) potentially used before declaration: retval ==
IERC721Receiver.onERC721Received.selector (contracts/CelestialsContract.sol#991)
Variable 'ERC721_checkOnERC721Received(address,address,uint256,bytes).reason (contracts/CelestialsContract.sol#992)' in
ERC721_checkOnERC721Received(address,address,uint256,bytes) (contracts/CelestialsContract.sol#983-1004) potentially used before declaration: reason.length == 0
(contracts/CelestialsContract.sol#993)
Variable 'ERC721_checkOnERC721Received(address,address,uint256,bytes).reason (contracts/CelestialsContract.sol#992)' in
ERC721_checkOnERC721Received(address,address,uint256,bytes) (contracts/CelestialsContract.sol#983-1004) potentially used before declaration: revert(uint256,uint256)(32 +
reason,mload(uint256)(reason)) (contracts/CelestialsContract.sol#997)

Address.isContract(address) (contracts/CelestialsContract.sol#353-363) uses assembly
- INLINE ASM (contracts/CelestialsContract.sol#359-361)
Address.verifyCallResult(bool,bytes,string) (contracts/CelestialsContract.sol#522-542) uses assembly
- INLINE ASM (contracts/CelestialsContract.sol#534-537)
ERC721_checkOnERC721Received(address,address,uint256,bytes) (contracts/CelestialsContract.sol#983-1004) uses assembly
- INLINE ASM (contracts/CelestialsContract.sol#996-998)

CelestialsContract.tokenURI(uint256) (contracts/CelestialsContract.sol#1329-1349) compares to a boolean constant:
-revealed == false (contracts/CelestialsContract.sol#1341)

Different versions of Solidity is used:
- Version used: ['>=0.7.0<0.9.0', '^0.8.0']
- ^0.8.0 (contracts/CelestialsContract.sol#44)
- ^0.8.0 (contracts/CelestialsContract.sol#67)
- ^0.8.0 (contracts/CelestialsContract.sol#207)
- ^0.8.0 (contracts/CelestialsContract.sol#233)
- ^0.8.0 (contracts/CelestialsContract.sol#261)
- ^0.8.0 (contracts/CelestialsContract.sol#330)
- ^0.8.0 (contracts/CelestialsContract.sol#549)
- ^0.8.0 (contracts/CelestialsContract.sol#577)
- ^0.8.0 (contracts/CelestialsContract.sol#603)
- ^0.8.0 (contracts/CelestialsContract.sol#626)
- ^0.8.0 (contracts/CelestialsContract.sol#1031)
- ^0.8.0 (contracts/CelestialsContract.sol#1193)
- >=0.7.0<0.9.0 (contracts/CelestialsContract.sol#1260)

ERC721Enumerable_removeTokenFromAllTokensEnumeration(uint256) (contracts/CelestialsContract.sol#1170-1188) has costly operations inside a loop:
- delete_allTokensIndex[tokenId] (contracts/CelestialsContract.sol#1186)
ERC721Enumerable_removeTokenFromAllTokensEnumeration(uint256) (contracts/CelestialsContract.sol#1170-1188) has costly operations inside a loop:
- _allTokens.pop() (contracts/CelestialsContract.sol#1187)
ERC721Enumerable_removeTokenFromOwnerEnumeration(address,uint256) (contracts/CelestialsContract.sol#1145-1163) has costly operations inside a loop:
- delete_ownedTokensIndex[tokenId] (contracts/CelestialsContract.sol#1161)

Address.functionCall(address,bytes) (contracts/CelestialsContract.sol#406-408) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/CelestialsContract.sol#416-422) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/CelestialsContract.sol#435-441) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/CelestialsContract.sol#449-460) is never used and should be removed
Address.functionDelegateCall(address,bytes) (contracts/CelestialsContract.sol#495-497) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/CelestialsContract.sol#505-514) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/CelestialsContract.sol#468-470) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/CelestialsContract.sol#478-487) is never used and should be removed
Address.sendValue(address,uint256) (contracts/CelestialsContract.sol#381-386) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (contracts/CelestialsContract.sol#522-542) is never used and should be removed
Context.msgData() (contracts/CelestialsContract.sol#619-621) is never used and should be removed
ERC721_baseURI() (contracts/CelestialsContract.sol#718-720) is never used and should be removed
ERC721_burn(uint256) (contracts/CelestialsContract.sol#918-930) is never used and should be removed
Strings.toHexString(uint256) (contracts/CelestialsContract.sol#297-308) is never used and should be removed
Strings.toHexString(uint256,uint256) (contracts/CelestialsContract.sol#313-323) is never used and should be removed

Pragma version^0.8.0 (contracts/CelestialsContract.sol#44) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#67) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#207) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#233) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#261) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#330) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#549) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#577) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#603) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#626) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#1031) allows old versions
Pragma version^0.8.0 (contracts/CelestialsContract.sol#1193) allows old versions
Pragma version>=0.7.0<0.9.0 (contracts/CelestialsContract.sol#1260) is too complex

Low level call in Address.sendValue(address,uint256) (contracts/CelestialsContract.sol#381-386):
- (success) = recipient.call{value: amount}() (contracts/CelestialsContract.sol#384)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/CelestialsContract.sol#449-460):
- (success,returndata) = target.call{value: value}(data) (contracts/CelestialsContract.sol#458)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/CelestialsContract.sol#478-487):
- (success,returndata) = target.staticcall(data) (contracts/CelestialsContract.sol#485)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/CelestialsContract.sol#505-514):
- (success,returndata) = target.delegatecall(data) (contracts/CelestialsContract.sol#512)
Low level call in CelestialsContract.withdraw() (contracts/CelestialsContract.sol#1383-1387):
- (os) = address(owner()).call{value: address(this).balance}() (contracts/CelestialsContract.sol#1385)

Parameter ERC721.safeTransferFrom(address,address,uint256,bytes).data (contracts/CelestialsContract.sol#795) is not in mixedCase
Parameter CelestialsContract.mint(uint256).mintAmount (contracts/CelestialsContract.sol#1292) is not in mixedCase
Parameter CelestialsContract.walletOfOwner(address).owner (contracts/CelestialsContract.sol#1316) is not in mixedCase
Parameter CelestialsContract.setCost(uint256).newCost (contracts/CelestialsContract.sol#1355) is not in mixedCase
Parameter CelestialsContract.setStopSupplyMintingAt(uint256).supplyStop (contracts/CelestialsContract.sol#1350) is not in mixedCase
Parameter CelestialsContract.setMaxMintAmount(uint256).newMaxMintAmount (contracts/CelestialsContract.sol#1363) is not in mixedCase
Parameter CelestialsContract.setNotRevealedURI(string).notRevealedURI (contracts/CelestialsContract.sol#1367) is not in mixedCase
Parameter CelestialsContract.setBaseURI(string).newBaseURI (contracts/CelestialsContract.sol#1371) is not in mixedCase
Parameter CelestialsContract.setBaseExtension(string).newBaseExtension (contracts/CelestialsContract.sol#1375) is not in mixedCase
Parameter CelestialsContract.pause(bool).state (contracts/CelestialsContract.sol#1379) is not in mixedCase

CelestialsContract.maxSupply (contracts/CelestialsContract.sol#1268) should be constant
```

```
name() should be declared external:
- ERC721.name() (contracts/CelestialsContract.sol#692-694)
symbol() should be declared external:
- ERC721.symbol() (contracts/CelestialsContract.sol#699-701)
tokenURI(uint256) should be declared external:
- CelestialsContract.tokenURI(uint256) (contracts/CelestialsContract.sol#1329-1349)
- ERC721.tokenURI(uint256) (contracts/CelestialsContract.sol#706-711)
approve(address,uint256) should be declared external:
- ERC721.approve(address,uint256) (contracts/CelestialsContract.sol#725-735)
setApprovalForAll(address,bool) should be declared external:
- ERC721.setApprovalForAll(address,bool) (contracts/CelestialsContract.sol#749-754)
transferFrom(address,address,uint256) should be declared external:
- ERC721.transferFrom(address,address,uint256) (contracts/CelestialsContract.sol#766-775)
safeTransferFrom(address,address,uint256) should be declared external:
- ERC721.safeTransferFrom(address,address,uint256) (contracts/CelestialsContract.sol#780-786)
tokenByIndex(uint256) should be declared external:
- ERC721Enumerable.tokenByIndex(uint256) (contracts/CelestialsContract.sol#1078-1081)
renounceOwnership() should be declared external:
- Ownable.renounceOwnership() (contracts/CelestialsContract.sol#1240-1242)
transferOwnership(address) should be declared external:
- Ownable.transferOwnership(address) (contracts/CelestialsContract.sol#1248-1251)
mint(uint256) should be declared external:
- CelestialsContract.mint(uint256) (contracts/CelestialsContract.sol#1292-1310)
viewBalanceUsed() should be declared external:
- CelestialsContract.viewBalanceUsed() (contracts/CelestialsContract.sol#1312-1314)
walletOfOwner(address) should be declared external:
- CelestialsContract.walletOfOwner(address) (contracts/CelestialsContract.sol#1316-1327)
reveal() should be declared external:
- CelestialsContract.reveal() (contracts/CelestialsContract.sol#1351-1353)
setCost(uint256) should be declared external:
- CelestialsContract.setCost(uint256) (contracts/CelestialsContract.sol#1355-1357)
setStopSupplyMintingAt(uint256) should be declared external:
- CelestialsContract.setStopSupplyMintingAt(uint256) (contracts/CelestialsContract.sol#1359-1361)
setMaxMintAmount(uint256) should be declared external:
- CelestialsContract.setMaxMintAmount(uint256) (contracts/CelestialsContract.sol#1363-1365)
setBaseExtension(string) should be declared external:
- CelestialsContract.setBaseExtension(string) (contracts/CelestialsContract.sol#1375-1377)
pause(bool) should be declared external:
- CelestialsContract.pause(bool) (contracts/CelestialsContract.sol#1379-1381)
withdraw() should be declared external:
- CelestialsContract.withdraw() (contracts/CelestialsContract.sol#1383-1387)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
mainnet:0x0858BDFC6c88137755049155426881dc2B04F89fc analyzed (13 contracts with 80 detectors), 81 result(s) found
```

Issues Categories

Total: [0 High, 0 Medium, 1 Low]

✓ High Severity Issues

No high severity issues found.

✓ Medium Severity Issues

No medium severity issues found.

✓ Low Severity Issues

1. In `CelestialsContract.walletOfOwner(address)` the variable `i` in `(contracts/CelestialsContract.sol#1323)` is a local variable never initialized.

Contract Interface:

1. The ERC721 interface is respected.

Other information:

1. The mint function is pausable by the owner. `(contracts/CelestialsContract.sol#1379)`
2. The reveal is only manageable by the owner. `(contracts/CelestialsContract.sol#1351)`
3. The urls of the tokens are well defined at the creation of the contract, but they can be modified later by the contract owner.
4. No deletion of tokens is possible. The `_burn` function is not used.
5. Limited to 4962 NFT, there will be no more.



Conclusion

Smart contracts do not contain high severity issues!

The contract is coded to respect the ERC721 structure.

The contract holder will be able to update the urls of the non-fungible tokens to potentially perform graphical updates.

No vulnerabilities were detected during our functionality tests.

A limit of 4962 NFT is set and cannot be changed later.

A maximum number of currencies per wallet is defined as well as a limit on the number of currencies, but this can be changed and will not be taken into account.

CheckDot note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by the Owner.