Smart Contract

Security Assessment

**For Cartoonistic
Puzzle Pieces
07 Jan 2023**

![Ascendant logo] Ascendant

Table of Contents

DISCLAIMER

Executive Summary

Severity	Found
🔴 High	1
🟠 Medium	3
🟡 Low	14
🟣 Informational	53
Total	71

We performed an independent technical audit to identify Smart Contracts uncertainties. This shall protect the code from illegitimate authorization attempts or external & internal threats of any type. This also ensures end-to-end proofing of the contract from frauds. The audit was performed semi-manually. We analyzed the Smart Contracts code line-by-line and used an automation tool to report any suspicious code.

The following tools were used:
- Truffle
- Remix IDE
- Slither

Overview

This report has been prepared for Cartoonistic Puzzle Pieces on the Ethereum network. Ascendant provides a user-centered examination of the smart contracts to look for vulnerabilities, logic errors or other issues from both an internal and external perspective.

Summary

Project Name	Cartoonistic Puzzle Pieces
Platform	Ethereum
Language	Solidity

Contracts Assessed

Name	Location
CartoonisticPuzzlePieces.sol	Goerli: 0x8938A5f8925AEbfE0Acba0A08a0A7a9BB53e6451
ERC721A.sol	In Cartoonistic Puzzle Pieces contract
Context.sol	In Cartoonistic Puzzle Pieces contract
Ownable.sol	In Cartoonistic Puzzle Pieces contract
ERC2981.sol	In Cartoonistic Puzzle Pieces contract
OperatorFilterer.sol	In Cartoonistic Puzzle Pieces contract

Name	Location
MerkleProof.sol	In Cartoonistic Puzzle Pieces contract
IERC721Metadata.sol	In Cartoonistic Puzzle Pieces contract
IERC165.sol	In Cartoonistic Puzzle Pieces contract
IERC721Enumerable.sol	In Cartoonistic Puzzle Pieces contract
IERC721.sol	In Cartoonistic Puzzle Pieces contract
IERC2981.sol	In Cartoonistic Puzzle Pieces contract
IERC721Receiver.sol	In Cartoonistic Puzzle Pieces contract
Address.sol	In Cartoonistic Puzzle Pieces contract
Strings.sol	In Cartoonistic Puzzle Pieces contract
IERC20.sol	In Cartoonistic Puzzle Pieces contract
DefaultOperatorFilterer.sol	In Cartoonistic Puzzle Pieces contract
IOperatorFilterRegistry.sol	In Cartoonistic Puzzle Pieces contract

Findings Summary

Severity	Found
🔴 High	1
🟠 Medium	3
🟡 Low	14
🟣 Informational	53
Total	71

Classification of Issues

🔴 High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency.
🟠 Medium	Bugs or issues that may be subject to exploit, though their impact is somewhat limited. Issues under this classification are recommended to be fixed as soon as possible.
🟡 Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
🟣 Informational	Consistency, syntax or style best practices, Generally pose a negligible level of risk, if any.

Manual Review

Issues Checking Status

Issue Description	Checking Status
Compiler errors	PASS
Race conditions and Reentrancy. Cross-function race conditions.	PASS
Possible delays in data delivery.	PASS
Oracle calls.	PASS
Front running.	PASS
Timestamp dependence.	PASS
Integer Overflow and Underflow.	PASS
DoS with Revert.	PASS
DoS with block gas limit.	PASS
Methods execution permissions.	PASS
Economy model of the contract.	PASS
The impact of the exchange rate on the logic.	PASS
Private user data leaks.	PASS
Malicious Event log.	PASS
Scoping and Declarations.	PASS
Uninitialized storage pointers.	PASS

Arithmetic accuracy.	PASS
Design Logic.	PASS
Cross-function race conditions.	PASS
Safe Open Zeppelin contracts implementation and usage.	PASS
Fallback function security.	PASS

Audit Findings

Severity	High
Contract	CartoonisticPuzzlePieces.sol
Description	Exposed baseURI
Code Snippet	string public baseURI = [REDACTED]
Recommendation	Writing the baseURI into the smart contract allows bad actors to obtain the location of the metadata and download all NFTs, even the ones that have not yet been minted. To prevent this, the baseURI variable should be set to private, and the constructor should accepted a string argument that will allow the owner to set the baseURI on deployment so the baseURI is visible to no one.
Status	

Audit Findings

Severity	Mediumx2
Contract	CartoonisticPuzzlePieces.sol
Description	Checks-Effects-Interactions
Code Snippet	function mint(... _safeMint(msg.sender, quantity); total_ps_count + quantity; minted[msg.sender] = minted[msg.sender] + quantity; function whitelistMint(... _safeMint(msg.sender, quantity); total_ps_count + quantity; minted[msg.sender] = minted[msg.sender] + quantity;
Recommendation	The mint functions currently all fail the checks-effects-interactions pattern, which requires that interactions (such as _safeMint) should come after state changes (updating the mapping), which exposes the function to Reentrancy attacks. total_ps_count + quantity; minted[msg.sender] = minted[msg.sender] + quantity should either precede _safeMint or a ReentrancyGuard should be added to these functions.
Status	

Audit Findings

Severity	Medium
Contract	CartoonisticPuzzlePieces.sol
Description	IERC20 is unnecessary/not utilized.
Code Snippet	N/A
Recommendation	CartoonisticPuzzlePieces is an ERC721A contract, which is a standard used for non-fungible tokens. ERC20 is a standard used for fungible tokens. Because their respective interfaces (IERC721 and IERC20) share some function names, in order to avoid any unwanted adverse effects when calling these functions, it is recommended to remove IERC20.
Status	

Audit Findings

Severity	Lowx2
Contract	CartoonisticPuzzlePieces.sol
Description	Quantity is an unneccesary parameter
Code Snippet	mint, whitelistMint functions
Recommendation	If the max per wallet is 1, there is no need for a quantity parameter for mint. Consider revising these functions to make the functions more gas efficient.
Status	

Audit Findings

Severity	Informational
Contract	CartoonisticPuzzlePieces.sol
Description	Minted mapping is redundant given that balanceOf function is inherited from IERC721
Code Snippet	1931: mapping(address => uint256) public minted;
Recommendation	ERC721A contracts inherit from IERC721, which includes a balanceOf function that checks the given address for the total minted by that address. In the interest of gas optimization, consider removing
Status	

Audit Findings

Severity	Informational(Multiple)
Contract	CartoonisticPuzzlePieces.sol
Description	Public functions that are used externally and not by the contract itself should be marked external
Code Snippet	N/A
Recommendation	Public functions generally consume more gas than external functions. Any functions that are not used internally should be marked external.
Status	

Functional Test Status

Function Name	Type/Return Type	Score
IOperatorFilterRegistry		
codeHashOf	external	PASS
copyEntriesOf	external	PASS
filteredCodeHashAt	external	PASS
filteredCodeHashes	external	PASS
filteredOperatorAt	external	PASS
filteredOperators	external	PASS
isCodeHashFiltered	external	PASS
isCodeHashOfFiltered	external	PASS
isOperatorAllowed	external	PASS
isOperatorFiltered	external	PASS
isRegistered	external	PASS
register	external	PASS
registerAndCopyEntries	external	PASS
registerAndSubscribe	external	PASS
subscribe	external	PASS
subscriberAt	external	PASS
subscribers	external	PASS

Function Name	Type/Return Type	Score
subscriptionOf	external	PASS
unregister	external	PASS
unsubscribe	external	PASS
updateCodeHash	external	PASS
updateCodeHashes	external	PASS
updateOperator	external	PASS
updateOperators	external	PASS
OperatorFilterer		
_checkFilterOperator	internal	PASS
IERC20		
allowance	external	PASS
approve	external	PASS
balanceOf	external	PASS
totalSupply	external	PASS
transfer	external	PASS
transferFrom	external	PASS

Function Name	Type/Return Type	Score
subscriptionOf	external	PASS
unregister	external	PASS
unsubscribe	external	PASS
updateCodeHash	external	PASS
updateCodeHashes	external	PASS
updateOperator	external	PASS
updateOperators	external	PASS
OperatorFilterer		
_checkFilterOperator	internal	PASS
IERC20		
allowance	external	PASS
approve	external	PASS
balanceOf	external	PASS
totalSupply	external	PASS
transfer	external	PASS
transferFrom	external	PASS

Function Name	Type/Return Type	Score
Context		
_msgData	internal	PASS
_msgSender	internal	PASS
Ownable		
_checkOwner	internal	PASS
transferOwnership	public	PASS
owner	public	PASS
renounceOwnership	public	PASS
Address		
functionCall	internal	PASS
functionCallWithValue	internal	PASS
functionDelegateCall	internal	PASS
functionStaticCall	internal	PASS
isContract	internal	PASS
sendValue	internal	PASS
verifyCallResult	internal	PASS
IERC721Receiver		
onERC721Received	external	PASS
ERC2981		

_deleteDefaultRoyalty	internal	PASS
_feeDenominator	internal	PASS
_resetTokenRoyalty	internal	PASS
_setDefaultRoyalty	internal	PASS
royaltyInfo	public	PASS
IERC721Metadata		
approve	external	PASS
balanceOf	external	PASS
getApproved	external	PASS
isApprovedForAll	external	PASS
ownerOf	external	PASS
safeTransferFrom	external	PASS
setApprovalForAll	external	PASS
transferFrom	external	PASS
IERC721		
approve	external	PASS
balanceOf	external	PASS
getApproved	external	PASS
isApprovedForAll	external	PASS
ownerOf	external	PASS
safeTransferFrom	external	PASS

setApprovalForAll	write/external	PASS
transferFrom	write/external	PASS
ERC721A		
_afterTokenTransfers	internal	PASS
_approve	private	PASS
_baseURI	internal	PASS
_beforeTokenTransfers	internal	PASS
_burn	internal	PASS
_checkOnERC721Received	private	PASS
_exists	internal	PASS
_mint	internal	PASS
_numberBurned	internal	PASS
_numberMinted	internal	PASS
_safeMint	internal	PASS
_transfer	private	PASS
approve	public	PASS
balanceOf	public	PASS
getApproved	public	PASS
isApprovedForAll	public	PASS
name	public	PASS
ownerOf	public	PASS
ownershipOf	internal	PASS

safeTransferFrom	public	PASS
setApprovalForAll	public	PASS
symbol	public	PASS
supportsInterface	public	PASS
tokenByIndex	public	PASS
tokenOfOwnerByIndex	public	PASS
tokenURI	public	PASS
totalSupply	public	PASS
transferFrom	public	PASS
MerkleProof		
verify	internal	PASS
CartoonisticPuzzlePieces		
_baseURI	internal	PASS
approve	public	PASS
dailyWithdraw	public	PASS
mint	public	PASS
refund	public	PASS
safeTransferFrom	public	PASS
setAccessRate	public	PASS
setApprovalForAll	public	PASS
setBaseURI	public	PASS
setContractURI	public	PASS

setFundReleasingPeriod	public	PASS
setMAX_Supply	public	PASS
setMax_per_wallet	public	PASS
setNotRevealedURI	public	PASS
setPublicSaleCost	public	PASS
setRoyaltyInfo	public	PASS
setTimeLimit	public	PASS
setTotal_ps_limit	public	PASS
setTotal_wl_limit	public	PASS
setWhitelistCost	public	PASS
setWhitelistSigner	public	PASS
toggleRefundActivity	public	PASS
toggleReveal	public	PASS
toggle_public_mint_status	public	PASS
toggle_wl_mint_status	public	PASS
tokenURI	public	PASS
transferFrom	public	PASS
whitelistMint	public	PASS

Omitted Results

Note: Any issues that have been omitted from this report have been deemed by the reviewing team as irrelevant, inapplicable, and/or negligible to the proper functioning of this contract. Thus, any omitted issues can be safely ignored.

Automated Review

Conclusion

The smart contracts reviewed in this audit contain no critical severity issues and all Medium to Low issues have either been corrected or acknowledged.

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 Owner.

Ascendant

Ascendant