



ONYX LABS

# Smart Contract Audit

Prepared For:  
The Orientalist

Date:  
11/14/22



# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>Summary</b>	<b>3</b>
<b>Overview</b>	<b>4</b>
<b>Findings</b>	<b>6</b>
ORIENT-001   Move to ERC721A	7
ORIENT-002   Use of CurrencyTransferLib/CollectClaimPrice	8
ORIENT-003   Value Splitting/Direct to Wallet	9
ORIENT-004   Bloated VerifyClaim Function	10
ORIENT-005   maxTotalSupply is not set by default.	11
ORIENT-006   Platform Fee	12
<b>Disclaimer</b>	<b>13</b>
<b>About</b>	<b>14</b>




## Summary

This report was prepared to summarize the findings of the audit performed for The Orientalist of their ERC-721 minting contract. The primary objective of the audit was to identify issues and vulnerabilities in the source code. This audit was performed utilizing Manual Review techniques and Static Analysis.

The audit consisted of:

- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Static Analysis utilizing Slither Static Analyzer.
- Unit testing specific functions.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Line-by-line manual review of the entire codebase by auditors.

The following report contains the results of the audit, and recommendations to guard against potential security threats and improve contract functionality.

A  next to a recommendation indicates that the recommendation was implemented prior to the completion of the audit.

We would like to thank the team from The Orientalist for their business and cooperation; their openness and communication made this audit a success.



# Overview

## Project Summary

Project Name	The Orientalist
Description	The Orientalist is an ERC721 NFT mint, built on the ThirdWeb “NFT Drop” contract. It allows for unlimited mint phases with variable pricing and claim amounts.
Blockchain	Ethereum
Language	Solidity
Codebase	Orientalist-ERC721.sol
Commit	N/A

## Audit Summary

Delivery Date	11/14/22
Audit Methodology	Static Analysis, Manual Review, Unit Testing

## Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Resolved
Critical	0	0	0	0	0
High	2	0	0	1	1
Medium	2	0	0	2	0
Low	1	0	0	1	0
Informational	1	0	0	1	0



## Audit Scope

ID	File
ORIENT	Orientalist-ERC721.sol



## Findings

ID	Title	Category	Severity	Status
ORIENT-001	Move to ERC721A	Optimization	High	
ORIENT-002	CurrencyTransferLib	Optimization	Medium	
ORIENT-003	Value Splitting/Direct to Wallet	Optimization	Medium	
ORIENT-004	Bloated VerifyClaim Function	Optimization	Low	
ORIENT-005	maxTotalSupply Unset by Default	Security	High	
ORIENT-006	Platform Fee	Information	Information	



## ORIENT-001 | Move to ERC721A

Category	Severity	Location	Status
Optimization	High	N/A	Acknowledged

### Description

The initial contract is constructed using a codebase from ThirdWeb, using open zeppelin ERC721/721 Enumerable standards. Thirdweb's codebase is very comprehensive, but designed in a way for maximum flexibility, sacrificing efficiency. ERC721A, developed by Chiru Labs, is currently one of the most optimized and widely used/audited implementations for ERC721 contracts. Using the ERC721A contract will dramatically improve the efficiency of the contract.

### Recommendation

Replace the TW/OZ ERC721/721 Enumerable with ERC721A and rework the necessary functions to fit.

### Action Taken

The audit was performed after contract deployment, and action could not be taken on this item.



## ORIENT-002 | Use of CurrencyTransferLib/CollectClaimPrice

Category	Severity	Location	Status
Optimization	Medium	N/A	Acknowledged

### Description

The contract utilizes the ThirdWeb library “CurrencyTransferLib” in the claim function. This library allows for non-native tokens to be used to pay for the mint. This is effective if needed, but is not needed in this instance and adds unnecessary bloat to the contract and reduces efficiency.

### Recommendation

Replace CurrencyTransferLib with a simple value check pattern.

### Action Taken

The audit was performed after contract deployment, and action could not be taken on this item.





## ORIENT-003 | Value Splitting/Direct to Wallet

Category	Severity	Location	Status
Optimization	Medium	N/A	Acknowledged

### Description

When tokens are minted, the contract directs the sent value directly to two wallets. This results in higher than normal transaction costs for the end user. It is more efficient to store the sent value on the contract, and withdraw/distribute it all at once.

### Recommendation

Remove logic to send ETH directly to external wallets, and replace it with a withdraw function that will send all of the collected funds to the contract owner and the other address listed in the payment splitter.

### Action Taken

The audit was performed after contract deployment, and action could not be taken on this item.



## ORIENT-004 | Bloated VerifyClaim Function

Category	Severity	Location	Status
Optimization	Medium	Line 372	Acknowledged

### Description

The function `verifyClaim` is used to ensure the correct parameters are met before minting tokens. This function contains unnecessary logic that reduces efficiency.

The `require` statement that verifies the correct currency is unnecessary/redundant and should be replaced with a simple value check pattern.

```
require(  
  _currency == currentClaimPhase.currency && _pricePerToken == currentClaimPhase.pricePerToken,  
  "invalid currency or price."  
);
```

Additionally, the logic that allows for `userSpecific` claim allowances is unnecessary. If this functionality is needed, it makes more sense to provide a separate function to facilitate.

### Recommendation

Remove redundant/unnecessary logic.

### Action Taken

The audit was performed after contract deployment, and action could not be taken on this item.



## ORIENT-005 | maxTotalSupply Unset by Default

Category	Severity	Location	Status
Security	High	Line 264	Resolved

### Description

The variable used to set a maximum supply of tokens for the collection, `maxTotalSupply`, is not set by default on contract deployment; if not manually set, the token is an open edition (no max supply). Additionally, the supply can be arbitrarily increased using the function `setMaxTotalSupply`. This is a serious concern for many collectors.

### Recommendation

Manually set the `maxTotalSupply`, or require it to be set in the constructor. Adjust `setMaxTotalSupply` so that the supply can be decreased, but not increased.

### Action Taken

`maxTotalSupply` was set using the `setMaxTotalSupply` function.



## ORIENT-006 | Platform Fee

Category	Severity	Location	Status
Information	Information	N/A	Acknowledged

### Description

The contract has embedded logic for a “platform fee” which sends a percentage of each sale to a separate wallet. Thirdweb no longer requires/asks for this to be done when using their contracts.

### Recommendation

If platform fee is being directed to Thirdweb, consider eliminating it.

### Action Taken

The audit was performed after contract deployment, and action could not be taken on this item.



## Disclaimer

The audit performed by Onyx Labs is intended to identify function weaknesses, potential security threats, and improve performance. It does not provide a guarantee of contract performance or the absence of any exploitable vulnerabilities. Any alterations to the source code provided could potentially invalidate the analysis provided in this audit report. Onyx Labs is not liable for any losses or damages incurred as a result of contract deployment and the proceeding occurrences.



## About

Onyx Labs is a full-service web3 development firm founded in December 2021. Their scope of expertise ranges from smart contract development for a multitude of applications, to web development and audits. By offering best-in-class service, Onyx Labs has helped many successful projects enter the web3 space, or provided assistance with their ongoing development needs.