HashEx
Blockchain Security

Sombra NFT

smart contracts
final audit report

October 2021

hashex.org

contact@hashex.org

Contents

1. Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice 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.

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 HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx 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 HashEx 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, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, 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 the 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. HashEx owns all copyright rights to the text, images, photographs, and other content provided in the following document. When using or sharing partly or in full, third parties must provide a direct link to the original document mentioning the author (hashex.org).

2. Overview

HashEx was commissioned by the Sombra team to perform an audit of their smart contract. The audit was conducted between October 1 and October 20, 2021.

The code was provided in an archive. In the files, there were no tests for audited contracts. Also, the code was provided without documentation.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts.
- Formally check the logic behind given smart contracts.

Information in this report should be used to understand the risk exposure of smart contracts, and as a guide to improving the security posture of smart contracts, by remediating the issues that were identified.

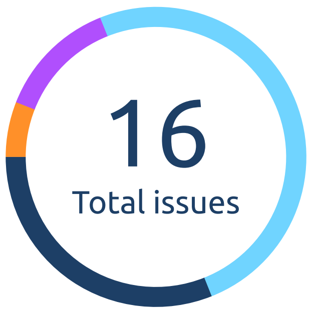
2.1 Summary

Project name	Sombra NFT
URL	https://www.sombra.app/
Platform	Binance Smart Chain
Language	Solidity

2.2 Contracts

Name	Address
SombraMarketplace	0x9B3b5968021eE3BcC88E6a75493BF0e44C694A12
SombraNFT	0x4c26fE419AE948a27D84f674Fc17ac82EBF32F51
Buyback	0x9B3b5968021eE3BcC88E6a75493BF0e44C694A12
WrappedETH	0x9B3b5968021eE3BcC88E6a75493BF0e44C694A12

3. Found issues



■	High	1 (6%)
■	Medium	2 (13%)
■	Low	8 (50%)
■	Informational	5 (31%)

SombraMarketplace

ID	Title	Severity	Status
01	Reentrancy	■ Medium	Resolved
02	Upper limit for bid time	■ Medium	Acknowledged
03	Event in try/catch block	■ Low	Resolved
04	Redundant argument	■ Low	Acknowledged
05	Structure assignment	■ Low	Acknowledged
06	No ERC721 compatibility checking	■ Low	Acknowledged
07	Enum usage	■ Informational	Acknowledged
08	Bad arguments	■ Informational	Acknowledged
09	Unindexed events	■ Informational	Acknowledged

10	Wrong interface	■ Informational	Acknowledged
11	Unused variable	■ Informational	Acknowledged

SombraNFT

ID	Title	Severity	Status
01	Require statements in view functions	■ Low	Acknowledged

Buyback

ID	Title	Severity	Status
01	Buyback path	■ High	Resolved
02	Visibility modifiers	■ Low	Acknowledged
03	Event in try/catch block	■ Low	Acknowledged

WrappedETH

ID	Title	Severity	Status
01	Event in try/catch block	■ Low	Resolved

4. Contracts

4.1 SombraMarketplace

4.1.1 Overview

The contract where purchases of NFT and sales take place.

4.1.2 Issues

01. Reentrancy

▪ Medium ✓ Resolved

The function closeAuction() L385 should have a reentrancy guard.

02. Upper limit for bid time

▪ Medium ! Acknowledged

In the function listItemOnAuction() L272 there is no checking on upper limit for the biddingTime argument.

03. Event in try/catch block

▪ Low ✓ Resolved

An appropriate event should be emitted in the catch section of the try/catch block in the buyFixedPriceItem(), placeBid() and closeAuction() functions.

04. Redundant argument

■ Low ⓘ Acknowledged

The argument tokenAddress In the function listItemOnAuction() is redundant. It always equals sombraNFTAddress.

Update

The requirement of tokenAddress == sombraNFTAddress was removed but the check in L268-269 remains, creating a hole in the contract's logic.

05. Structure assignment

■ Low ⓘ Acknowledged

The code on L332, 382, 409 is not gas-wise because only one variable in the structure is changed, but the whole structure is rewritten in global storage.

06. No ERC721 compatibility checking

■ Low ⓘ Acknowledged

In the function placeBid() L341 there is no check on ERC721 compatibility of msg.sender.

07. Enum usage

- Informational ⓘ Acknowledged

Contract variables ON_MARKET, SOLD and CANCELLED may be combined in a single enum.

08. Bad arguments

- Informational ⓘ Acknowledged

The arguments purchasePrice and startingBidPrice in the listItemOnAuction() function, can both be non-zero at the same time. It's similar in the reducePrice() function with the arguments reducedPrice and reducedBidPrice.

09. Unindexed events

- Informational ⓘ Acknowledged

Events should contain indexed parameters for better usability.

10. Wrong interface

- Informational ⓘ Acknowledged

The interface of SombraNFT contains an error. minter() function in SombraNFT should be marked as 'view'. Also, in SombraMarketplace functions minter() L188 and isMinter() could be marked as 'view' too.

11. Unused variable

▪ Informational ⓘ Acknowledged

The variable auctionStartTime in the structure MarketItem is not used in the contract. Using it only for the front-end is not a gas-friendly option as the event mechanism is designed specifically for that purpose.

4.2 SombraNFT

4.2.1 Overview

This is the contract under ERC721 standard. Anyone can mint new id of this nft.

4.2.2 Issues

01. Require statements in view functions

▪ Low ⓘ Acknowledged

It is an incorrect practice to place require statements in view functions. Explorers may show unrealistic return values of these functions. All the require checks could be moved to the corresponding non-view functions.

4.3 Buyback

4.3.1 Overview

Functionality for buyback tokens from DEX.

4.3.2 Issues

01. Buyback path

■ High ✓ Resolved

When buyback proceeds, the path is set to [WETH, SombraMarketplace address] but SombraMarketplace contract is not an ERC20 token and there cannot be a pair with this contract. Maybe in the function buybackSombra() the variable address(this) should be replaced with an address of the Sombra ERC20 token.

02. Visibility modifiers

■ Low ⓘ Acknowledged

The variables WETH and ethToBuybackWith have no visibility modifiers.

03. Event in try/catch block

■ Low ⓘ Acknowledged

An appropriate event should be emitted in the catch section of the try/catch block in the swapETHForTokens() function.

4.4 WrappedETH

4.4.1 Overview

Functionality for recording the balance of the users in the ETH and sending this balance to the account.

4.4.2 Issues

01. Event in try/catch block

▪ Low ✓ Resolved

An appropriate event should be emitted in the catch section of the try/catch block in the rewardBNBToUserAndClaim() function.

5. Conclusion

One high severity issue was found. We strongly recommend adding tests with coverage of at least 90%, before the deployment to the mainnet.

This audit includes recommendations on the code improving and preventing potential attacks.

The Sombra team has responded to this report. Most of the issues have been fixed including the high severity one. The updated contracts are deployed to the Binance Smart Chain: SombraNFT and SombraMarketplace.

Appendix A. Issues' severity classification

Critical. Issues that may cause an unlimited loss of funds or entirely break the contract workflow. Malicious code (including malicious modification of libraries) is also treated as a critical severity issue. These issues must be fixed before deployments or fixed in already running projects as soon as possible.

High. Issues that may lead to a limited loss of funds, break interaction with users, or other contracts under specific conditions. Also, issues in a smart contract, that allow a privileged account the ability to steal or block other users' funds.

Medium. Issues that do not lead to a loss of funds directly, but break the contract logic. May lead to failures in contracts operation.

Low. Issues that are of a non-optimal code character, for instance, gas optimization tips, unused variables, errors in messages.

Informational. Issues that do not impact the contract operation. Usually, informational severity issues are related to code best practices, e.g. style guide.

Appendix B. List of examined issue types

- Business logic overview
- Functionality checks
- Following best practices
- Access control and authorization
- Reentrancy attacks
- Front-run attacks
- DoS with (unexpected) revert
- DoS with block gas limit
- Transaction-ordering dependence
- ERC/BEP and other standards violation
- Unchecked math
- Implicit visibility levels
- Excessive gas usage
- Timestamp dependence
- Forcibly sending ether to a contract
- Weak sources of randomness
- Shadowing state variables
- Usage of deprecated code

HashEx

Blockchain Security