ScapeNftCollection

Date	May 2024

1 Executive Summary

This report presents the results of our engagement with the **Scape team** to review **ScapeNftCollection**.

The review was conducted over one week, from **29/04/2024** to **6/05/2024**, by **Arturo Roura** and **François Legué**.

The engagement consisted on a modified version of a codebase that was already reviewed by the Diligence team. The new engagement has centered around the `ScapeNftCollection` contract, formerly recognized as `ScapeFoundingCitizens`. This endeavor introduces enhanced features leveraging newly integrated whitelisting capabilities, alongside the utilization of `ERC721-C` contracts sourced from LimitBreak.

The ScapeNftCollection contract mostly integrates rigorously vetted community standards, augmenting its robustness. The codebase adheres to principles of clarity and conciseness, thereby minimizing potential vulnerabilities. The implementation predominantly employs access controls to safeguard critical functions. Moreover, functions lacking such controls typically possess minimal arguments, mitigating the likelihood of extreme edge cases.

While the contract incorporates additional functionalities from `ERC721-C`, it's important to note that the standard itself was not within the primary scope of this engagement. Nonetheless, a conscientious endeavor has been made to grasp pertinent security considerations associated with this standard.

2 Scope

While the previous engagement focused on reviewing commit hash `ca36861006c612f1665a33a563afb376a0011e7b`, this review focused on the commit hash `df499526778b51daa69f5eb69ce22fc3c9d7467d`.

The list of files in scope can be found in the Appendix.

2.1 Objectives

Together with the **Scape** team, we identified the following priorities for our review:

1. Correctness of the implementation, consistent with the intended functionality and without unintended edge cases.
2. Identify known vulnerabilities particular to smart contract systems, as outlined in our Smart Contract Best Practices, and the Smart Contract Weakness Classification Registry.
3. Correctness in the added functionality regarding Merkle trees and minting amounts.
4. While `ERC721-C` is out of scope, efforts have been invested in ensuring the accuracy of its implementation.

3 System Overview

The ScapeNftCollection contract's main logic focuses on the correct minting and distribution of the ERC721-C.

This is based on two main periods where users can mint their own NFT, and the surrounding time periods. The period timestamps are validated upon contract creation in the _validateMintSchedule function. The periods are:

- Pre-whitelist minting period: Anything prior to whitelistStartTime will revert. No one is supposed to be able to mint during this period.
- Whitelist minting period: From whitelistStartTime to whitelistEndTIme(exclusive), users with an eligible merkle proof can mint up to the _whitelistedAmount, which is hashed in their proof.
- Intermediate period: While not enforced in the contract, a period between the whitelistedEndTime and the publicSaleStartTime can exist. In this period no minting could be executed.
- Public minting period: A period where anyone can mint. The period is originally set from publicSaleStartTime inclusive to publicSaleEndTime exclusive, but can be triggered sooner under certain conditions.
    - The conditions to trigger the public minting period sooner are:
        - All whitelisted tokens have been minted
        - autoPublicMintSwitchover has to be set to true by the owner
- Sale finished period: From publicSaleEndTime inclusive onwards, during this period the sale has ended and only the owner can mint tokens until maxSupply is reached.

Inheritance Structure:

The ERC721-C is used to enforce royalties, relying on another contract called "transferValidator" to validate transfers. To do so, ERC721C overrides the "_beforeTokenTransfer" hook from the original ERC721 implementation, which is called upon every transfer, and calls "_validateBeforeTransfer".

ERC721C inherits from “CreatorTokenBase”, the contract where the main logic that keeps track of the “transferValidator” and makes calls to it to modify the “TRANSFER_SECURITY_LEVEL”, “OperatorWhitelistOfCollection” and “PermittedContractReceiverAllowlistOfCollection”, variables which defines if a transfer will or won’t be valid. It’s in this contract where the “_preValidateTransfer” is defined, which is the function called on every transfer by the previously mentioned “_validateBeforeTransfer”. _preValidateTransfer will call registered “transferValidator” to check if the transfer is valid.

Based on Seaport v1.6, the expected use of ERC721-C is for the “transferValidator” to act as a “zone”. To enforce royalties, transfers will only be made by the Seaport contract, or any other whitelisted marketplace contract address. The Seaport contract during the sale of the NFT will call the “transferValidator” contract which will attest to the inclusion of creator fees on the associated Seaport order, in this case, the fees specified in “royaltyInfo()” based on eip-2981. If the requested criteria is matched, the future transfer is flagged as valid. On the subsequent transfer during the order fulfillment, the ERC721-C will call the “transferValidator” to see if the transfer is flagged as valid, and will revert otherwise.

4 Security Specification

This section describes, **from a security perspective**, the expected behavior of the system under audit. It is not a substitute for documentation. The purpose of this section is to identify specific security properties that were validated by the audit team.

4.1 Actors

The relevant actors are listed below with their respective abilities:

1. Contract Owner → Admin operations
2. Contract Manager → Limited to mint pausing and withdrawal to owner address
3. Whitelisted Account → Limited to whitelisted minting
4. Public Account → Regular minting, all standard and read-only public functionalities

4.2 Security Properties

The following is a non-exhaustive list of security properties that were verified in this audit on each function:

- `whitelistedMint()`:
  - The invariants that should uphold are:
    - the tokens minted should never surpass `maxWhitelistSupply`
    - each user should only be able to mint a maximum of `whitelistedAmount` tokens
    - The amount sent while minting should equate to the amount of tokens minted times the minting price
  - `tokenIndexId` isn’t checked against `maxSupply` in this function, but it shouldn’t be a problem since `maxSupply` should be greater than `maxWhitelistSupply`.
  - The function’s security is bolstered by utilizing `msg.sender` for authentication, ensuring the precise amount sent is verified, and executing the process with clarity and precision.
- `publicMint()`:
  - The invariants that should uphold are:
    - the tokens minted should never surpass `maxSupply`
    - the tokens minted in a single tx should never surpass `MAX_PUBLIC_MINT_PER_TRANSACTION`
    - The amount sent while minting should equate to the amount of tokens minted times the minting price
  - Despite the added complexity of enabling public minting before `publicStateStartTime`, the inclusion of the `autoPublicMintSwitchover` boolean provides an additional layer of security.
- `managerOrOwner` functions:
  - The contract manager can withdraw tokens to the owner and pause the contract’s functionality.
- `onlyOwner` functions:
  - The contract’s most critical state variables and execution rely on the “onlyOwner” modifier. The owner has the power to change prices, periods, withdraw all native tokens, validate `autoPublicMintSwitchover`, mint all remaining tokens after the sale has ended, change the manager and ultimately pause and unpause the contract functionality.
  - Furthermore, based on the overridden “_requireCallerIsContractOwner()” function to act as the “onlyOwner” modifier, the owner is able to execute the inherited functions from the “CreatorTokenBase” contract, which are critical for the correct functioning of ERC721-C.

Notable mentions:

- The `_requireCallerIsContractOwner()` has been checked and works as expected based on the tests. It is a critical part of the functionality of ERC721-C, since it will be the main access control from the key functions in `CreatorTokenBase.sol`. These access controls can alter the functionality of a regular transfer to make censor it.
- The `_burn()` function overrides it’s original implementation applying access controls and modifying the custom `totalSupply`.

5 Findings

Each issue has an assigned severity:

- Minor issues are subjective in nature. They are typically suggestions around best practices or readability. Code maintainers should use their own judgment as to whether to address such issues.
- Medium issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.

- Major issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
- Critical issues are directly exploitable security vulnerabilities that need to be fixed.

5.1 Missing Testing for ERC721-C Enforced Royalties Minor Acknowledged

Resolution
The client decided to not extend the test suite for external ERC721C library

The testing framework adequately scrutinizes the foundational functionalities within the contract ecosystem, affirming their expected operational efficacy. However, in the context of `ERC721-C` enforced royalties, tests merely confirm the successful configuration of variables within `CreatorTokenBase.sol` (a dependency contract from `ERC721-C`) to be set to security level 1.

While our examination of the `ERC721-C` implementation revealed no issues regarding its compatibility with the codebase, the testing coverage should incorporate marketplace scenarios, thus ensuring the rigorous enforcement of royalties.

Additionally, it's worth noting that the enforced royalties associated with LimitBreak's security levels 0, 1, and 2 can be potentially bypassed through the use of an intermediary contract. On the other hand, security levels 3, 4, 5 and 6 require the receiver to have no code or to be an EOA and therefore aren't subject to this possible circumvention.

Appendix 1 - Files in Scope

This audit covered the following files:

File	SHA-1 hash
ScapeNftCollection/contracts/ScapeNftCollection.sol	`712ffa5a7982cbb188d33d886fbcfba3cb7a860c`

Appendix 2 - Disclosure

Consensys Diligence ("CD") typically receives compensation from one or more clients (the "Clients") for performing the analysis contained in these reports (the "Reports"). The Reports may be distributed through other means, including via Consensys publications and other distributions.

The Reports are not an endorsement or indictment of any particular project or team, and the Reports do not guarantee the security of any particular project. This Report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any third party in any respect, including regarding the bug-free nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. No third party should rely on the Reports in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. Specifically, for the avoidance of doubt, this Report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. CD owes no duty to any third party by virtue of publishing these Reports.

A.2.1 Purpose of Reports

The Reports and the analysis described therein are created solely for Clients and published with their consent. The scope of our review is limited to a review of code and only the code we note as being within the scope of our review within this report. Any Solidity code itself presents unique and unquantifiable risks as the Solidity language itself remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond specified code that could present security risks. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. In some instances, we may perform penetration testing or infrastructure assessments depending on the scope of the particular engagement.

CD makes the Reports available to parties other than the Clients (i.e., "third parties") on its website. CD hopes that by making these analyses publicly available, it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.

A.2.2 Links to Other Web Sites from This Web Site

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Consensys and CD. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Consensys and CD are not responsible for the content or operation of such Web sites, and that Consensys and CD shall have no liability to you or any other person or entity for the use of third party Web sites. Except as described below, a hyperlink from this web Site to another web site does not imply or mean that Consensys and CD endorses the content on that Web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the Reports. Consensys and CD assumes no responsibility for the use of third-party software on the Web Site and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

A.2.3 Timeliness of Content

The content contained in the Reports is current as of the date appearing on the Report and is subject to change without notice unless indicated otherwise, by Consensys and CD.