Scape

Date	March 2024
Auditors	Rai Yang, François Legué

1 Executive Summary

This report presents the results of our engagement with **Scape** team to review **Scape300** and **ScapeFoundingCitizens** NFT contracts.

The review was conducted by **Rai Yang** and **François Legué**. It took place between **March 4th, 2024**, and **March 8th, 2024**.

The **Scape300** and **ScapeFoundingCitizens** contracts represent two NFT collections. Claiming of NFTs is reserved to whitelisted users for the **Scape300** contract. The **ScapeFoundingCitizens** contract minting schedule is composed of two phases:

- the whitelisted minting phase;
- the public minting phase.

2 Scope

Our review focused on the commit hash `ca36861006c612f1665a33a563afb376a0011e7b`. All findings that we identified have been fixed or acknowledged in the version with commit hash `78740502e8af14448640d9eea127976525cd0ed0`. 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 of the usage of Merkle trees

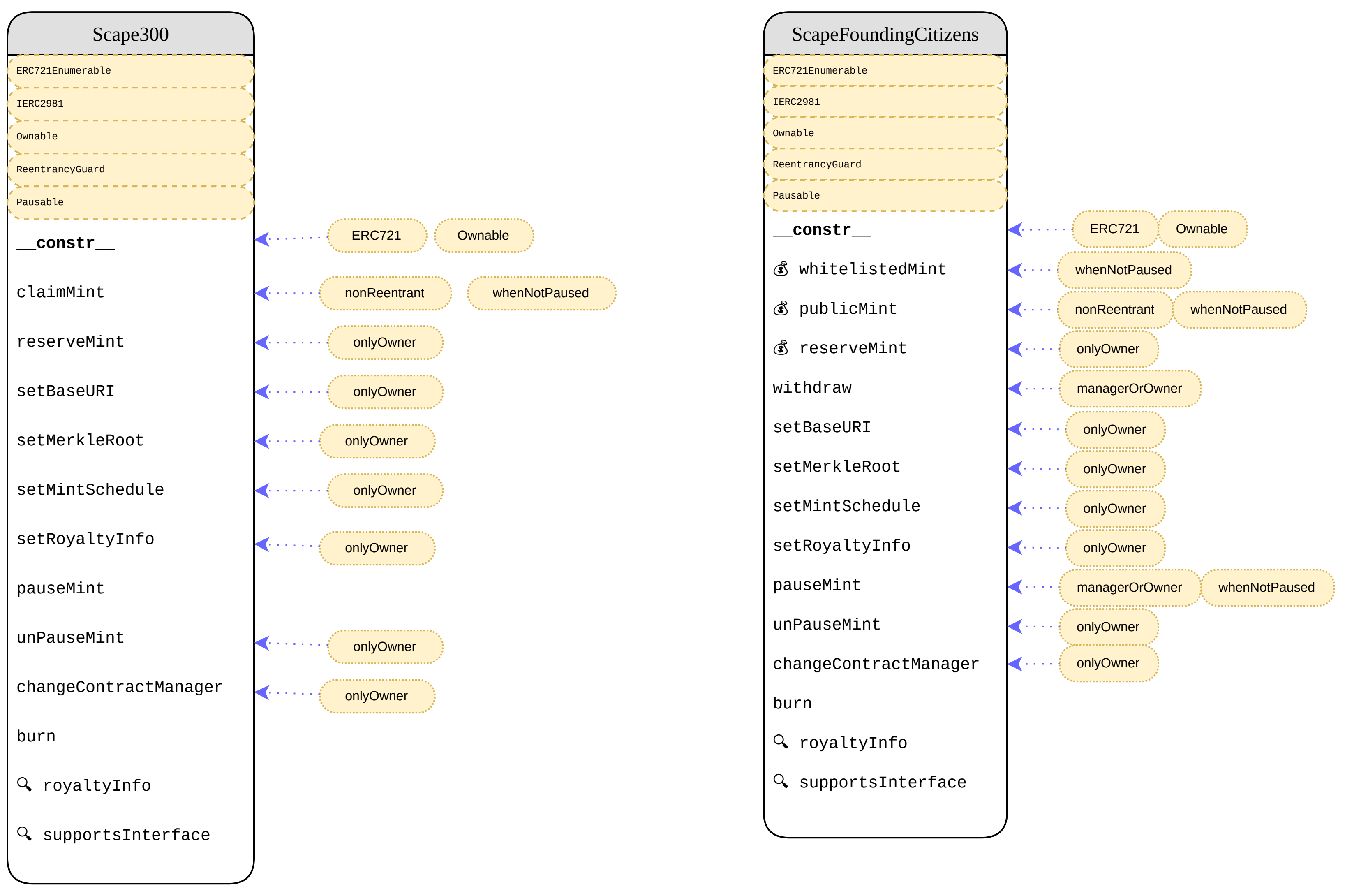
3 System Overview

The **Scape300** and the **ScapeFoundingCitizens** contracts can be used to mint NFTs. The whitelisting of the users authorized to mint an NFT is based on the usage of Merkle trees. A Merkle tree, also known as a hash tree, is a data structure that plays a crucial role in efficiently verifying that the user submits a valid proof demonstrating that he is part of the whitelist.

The Merkle tree is often used for whitelisting because it is:

- efficient: when checking if a user is whitelisted, only a small number of hashes from the user's data hash to the root of the Merkle tree have to be computed.
- security/integrity: as it relies on a hashing algorithm and one of its fundamental properties (non-reversible), this structure ensures that the whitelist cannot be tampered with.
- privacy: for the same reason from the above point, the whitelisted addresses are kept private until the minting.

These two contracts are based on the `ERC-721` standard, its Enumerable extension, and the Royalty standard (`ERC-2981`).



Scape300 and ScapeFoundingCitizens contracts overview

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:

- **Smart contract owner** - The owner has several administrative permissions including changing the base URI of the tokens in the collection, changing the whitelist Merkle Root, setting the minting schedule, setting the royalty info, pausing or un-pausing the minting operation, and changing the contract manager. The owner also has the ability to perform reserve minting.
- **Contract Manager** - The contract manager is defined by the owner of the contract and has the ability to pause the minting.
- **Whitelisted Users** - Whitelisted users who are allowed to claim and mint NFTs.
- **Public Users** - This actor is specific to `ScapeFoundingCitizens` contract. Once the whitelisting phase is over, any account is allowed to mint NFTs the remaining NFTs until the `MAX_SUPPLY` is reached or the public minting phase has ended.

4.2 Trust Model

In any system, it's important to identify what trust is expected/required between various actors. For this audit, we established the following trust model:

- The **Smart contract owner** is trusted to sets all the details of the NFTs collections correctly and the owner account is owned by a secure Multisig. It is to be noted that this actor is able to:
  - deploy the token contract;
  - update the minting schedule;
  - update the Merkle root which is the core component of the whitelisting mechanism;
  - update the Royalty info (beneficiary address and royalty rate),
  - withdraw the minting fee from the `ScapeFoundingCitizens` contract
- The Contract Manager performs management tasks correctly.
- The Merkle tree root is trusted to be correctly generated. Tests currently generate the Merkle tree root by relying on the OpenZeppelin library which contains multiple mitigations against known attacks.
- The total amount of whitelisted NFTs is also trusted to be under the `MAX_SUPPLY` amount.

4.3 Security Properties

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

- Correct role based access control
- Correct Merkle tree leaf hashing algorithm: The current leaf hashing algorithm is based on OpenZeppelin's implementation, using twice the `keccak256` hash function. This protects against second pre-image attacks.
- Reentrancy attack on minting and claiming

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 Unnecessary `payable` Attribute for `reserveMint()` Function in `ScapeFoundingCitizens` Contract Minor Acknowledged

Resolution

Scape team has acknowledged this issue and chose to keep the `payable` keyword to save gas on each call.

Description

The `reserveMint()` function allows the `owner` of the NFT contract to mint the remaining NFTs in the collection after an expiration date (`claimingEndTime` for `Scape300` and `publicSaleEndTime` for `ScapeFoundingCitizens`).

This function is unnecessarily marked as `payable` in the `ScapeFoundingCitizens` contract.

ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol:L202

```
function reserveMint(uint256 _amount) external payable onlyOwner {
```

The incorrect use of the `payable` keyword does not pose a major threat but could lead to confusion and accidental transfers of Ether to a function not meant to receive it.

Recommendation

Remove the `payable` attribute from the `reserveMint()` function declaration.

5.2 Missing Events on Important State Changes Minor ✓ Fixed

Description

The `setBaseURI()` and `setMintingSchedule()` state-changing functions do not emit an event. Events help to track state changes of smart contracts and make it easier for off-chain tools to track critical updates. It's always better to emit events containing relevant updates.

ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol:L243-L245

```
function setBaseURI(string memory _tokenBaseURI) external onlyOwner {
  baseURI = _tokenBaseURI;
}
```

Scape300/contracts/Scape300.sol:L170-L172

```
function setBaseURI(string memory _tokenBaseURI) external onlyOwner {
  baseURI = _tokenBaseURI;
}
```

Scape300/contracts/Scape300.sol:L185-L188

```
function setMintSchedule(MintSchedule memory _mintSchedule) external onlyOwner {
  _validateMintSchedule(_mintSchedule);
  mintSchedule = _mintSchedule;
}
```

ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol:L258-L261

```
function setMintSchedule(MintSchedule memory _mintSchedule) external onlyOwner {
  _validateMintSchedule(_mintSchedule);
  mintSchedule = _mintSchedule;
}
```

Recommendation

Consider emitting an event on important state change.

5.3 Enhancements Required for Complete Test Suite Coverage ✓ Fixed

Description

The provided code base contains a test suite. Tests are crucial in the development cycle and help ensure that every part of the code works as expected. The current coverage is currently very high and almost complete.

However, it could be improved further to cover the following cases:

- The `publicMint()` function should revert with `PublicMintNotAllowed()` when the `block.timestamp` is over the `publicSaleEndTime`

ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol:L164-L167

```
  (tokenIdIndex < MAX_WHITELIST_SUPPLY && block.timestamp < mintSchedule.publicSaleStartTime) ||
  block.timestamp >= mintSchedule.publicSaleEndTime
) {
  revert PublicMintNotAllowed();
```

- The `_validateMintSchedule()` function should revert with `MintScheduleInvalid()` when `whitelistStartTime` is equal to `0` or when `publicSaleStartTime` is greater than or equal to the `publicSaleEndtime`

ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol:L348-L357

```
function _validateMintSchedule(MintSchedule memory _mintSchedule) internal pure {
  if (
    _mintSchedule.whitelistStartTime == 0 ||
    _mintSchedule.whitelistStartTime >= _mintSchedule.whitelistEndTime ||
    _mintSchedule.publicSaleStartTime >= _mintSchedule.publicSaleEndTime ||
    _mintSchedule.whitelistEndTime > _mintSchedule.publicSaleStartTime
  ) {
    revert MintScheduleInvalid();
  }
}
```

- The `withdraw()` function should revert with `WithdrawalFailed()` error when the low-level `call` fails

ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol:L233-L238

```
function withdraw() external managerOrOwner {
  (bool success, ) = owner().call{ value: address(this).balance }("");
  if (!success) {
    revert WithdrawalFailed();
  }
}
```

- The `claimMint()` in `Scape300` contract function should revert with the `EnforcedPause` error when a whitelisted user tries to mint an NFT when the contract is paused
- The current test named `Owner account SHOULD NOT BE able to MINT tokens if max supply reached` contains multiple errors

ScapeFoundingCitizens/test/ReserveMint.test.ts:L123-L126

```
expect(contractOwnerInstance.reserveMint(mintAmount)).to.be.revertedWithCustomError(
  contractOwnerInstance,
  'MaxSupplyReached'
);
```

This test wrongly passes because the transaction associated with the `reserveMint()` is not awaited. Moreover, the defined `maxSupply` does not match the value specified for the contract deployment (`10000`).

ScapeFoundingCitizens/test/helpers/Utils.ts:L24-L35

```
export const scapeFoundingCitizensConfig = {
  mintPrice: ethers.parseEther('0.2'),
  maxSupply: 10000,
  maxWhitelistMint: 7000,
  maxPublicMintPerWallet: 2,
  // Note: Please note that ending / is important
  initialBaseUri: 'ipfs://INITIAL/',
  finalBaseUri: 'ipfs://FINAL/',
  initialRoyaltyBasisPoints: 700,
  name: 'SCAPE: Founding Citizens',
  symbol: 'SCPCIT',
};
```

Recommendation

Consider updating the current test suite to cover these specific cases to enhance the coverage.

Appendix 1 - Files in Scope

This audit covered the following files:

File	SHA-1 hash
Scape300/contracts/Scape300.sol	`d4a1104e5962b0be84b61e2d7b8309c409730f2c`
ScapeFoundingCitizens/contracts/ScapeFoundingCitizens.sol	`1b0c42195fa1e6d9fe7fccd377dc42cc61776731`

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.