Smart Contract
Security Audit
V1

IncubateX NFT Smart Contract

28/6/2022

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

–

Table of Contents

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Project Information

- **Platform**: Polygon Network
- **Contract Address**: 0x846ea1212276F9db7E26218c026E66eB00c05f3f
- **Code:**

https://ropsten.etherscan.io/address/0xb56a656907ba4ea34daa106412b52253b8ea0bd2#code

- **The Code:**

https://mumbai.polygonscan.com/address/0x489bF55C961Fb94d732ED6C7Af97f343603087aC#code

NFT Information

- Name: IX
- Total Supply: 22222
- Holders:
- Total transactions:

Contracts address deployed to test net (Polygon Network)

IncubateX NFT Smart contract on Polygon test net to test write functions by the auditor.

https://mumbai.polygonscan.com/address/0x846ea1212276f9db7e26218c026e66eb00c05f3f

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Secured**.

Well Secured	
Secured	✓
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 3 low, 0 very low-level issues and 0 note in all solidity files of the contract

The files:

IncubateX.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
IncubateX.sol	**b6df8bcfde41c117b1d9f7a25a159d0d0d4161591461bc26bc5c92abe456f789**	0x846ea1212276F9db7E26218c026E66eB00c05f3f

- Contract: IncubateX
- Inherit: ERC721Enumerable, Ownable
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
name	✓	Read / public	**Passed**
symbol	✓	Read / public	**Passed**
saleIsActive	✓	Read / public	**Passed**
supportsInterface	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
genesisCount	✓	Read / public	**Passed**
getMintedGenesisTokens	✓	Read / public	**Passed**
getApprovedForAll	✓	Read / public	**Passed**
ownerOf	✓	Read / public	**Passed**
getApproved	✓	Read / public	**Passed**

tokenURI	✓	Read / public	**Passed**
tokenByIndex	✓	Read / public	**Passed**
tokenOfOwnerByIndex	✓	Read / public	**Passed**
PRICE_PER_NFT_Genesis	✓	Read / public	**Passed**
PRICE_PER_NFT_Meta	✓	Read / public	**Passed**
verifyUserisWL	✓	Read / public	**Passed**
getMintedMetaCapsule	✓	Read / public	**Passed**
addUserWL	✓	Write / public	**Passed**
approve	✓	Write / public	**Passed**
safeTransferFrom	✓	Write / public	**Passed**
safeTransferFrom	✓	Write / public	**Passed**
mintGenesisNft	✓	Write / payable	**Passed**
mintMetaNft	✓	Write / payable	**Passed**
transferOwnership	✓	Write / public	**Passed**
setApprovalForAll	✓	Write / public	**Passed**
transferFrom	✓	Write / public	**Passed**
withdrawAll	✓	Write / public	**Passed**
setMetaBaseURI	✓	Write / public	**Passed**
setGenesisPrice	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
withdraw	✓	Write / public	**Passed**
setMetaPrice	✓	Write / public	**Passed**
setSaleIsActive	✓	Write / public	**Passed**

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	**Passed**
2	Race conditions and Reentrancy. Cross-function race conditions.	**Passed**
3	Possible delays in data delivery.	**Passed**
4	Oracle calls.	**Passed**
5	Design Logic.	**Passed**
6	Timestamp dependence.	**Passed**
7	Integer Overflow and Underflow.	**Passed**
8	DoS with Revert.	**Passed**
9	DoS with block gas limit.	**Passed**
10	Methods execution permissions.	**Passed**
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	**Passed**
12	The impact of the exchange rate on the logic.	**Passed**
13	Private user data leaks.	**Passed**
14	Malicious Event log.	**Passed**
15	Scoping and Declarations.	**Passed**
16	Uninitialized storage pointers.	**Passed**
17	Arithmetic accuracy.	**Passed**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can’t lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can’t have significant impact on execution
Note	Lowest-level vulnerabilities, code style violations and info statements can’t affect smart contract execution and can be ignored.

Audit Findings

Critical:

#Logic errors

Description
According to the business plan the team had sent to the auditor, the auditor had found these logic errors on the smart contract.

The Error	Description
Max Supply	The team want(11,111 IncubateX Genesis Membership & 11,111 Meta Access Capsule NFTs available for mint on mint day, for a total of 22,222 NFTs) The Max Supply is Unlimited.
Whitelist error	The team want (25% of the IncubateX NFTs will be available for whitelisted community members during the presale, 2,777 Genesis Membership NFTs and 2,777 Meta Access Capsule NFTs) The error any one can mint the IX NFT at whitelist stage without being in the whitelist and there aren't any limits for minting at whitelist stage.
Secondary sales	If a member decides to independently sell their IncubateX NFT on the secondary market, they would be transferring the membership utility of that NFT to the new NFT holder. There will be a 10% royalty for secondary market sales for both the Genesis Membership NFT & Meta Access Capsule NFT. Revenue from royalties should go to to the same wallet as the project wallet for mint. The error this is normal NFT smart contract not royalty NFT contract.
Limits	Each wallet address is able to mint 10 of each of the NFTs; 10 IncubateX Genesis Membership NFTs & 10 Meta Access Capsules The error any address can hold any numbers there aren't any limits of holding IX Genesis or IX Meta.
Updates	This project is a private company project and we need to be able to make updates to the code later on if necessary.

	The error after deploying the team can only change a few things on the code using the write functions like the price, baseURI, and withdraw the funds but they can't update the smart contract.

Status: Closed. Fixed in version 2.

High:

No High severity vulnerabilities were found.

Medium:

No Medium severity vulnerabilities were found.

Low:

#Missing zero address validation

Description

When the investor wants to add his address to the whitelist, he has to check for the zero address to make, he didn't add the burn address. Otherwise, the investor will lose some funds as gas.

```
function addUserWL(address _addressToWhitelist) public  {
         whitelistedAddresses[_addressToWhitelist] = true;
     }
```

Remediation

Use the require statement to check for zero addresses.

Status: Closed. Fixed in version 2.

#Pragam version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.14 instead of ^0.8.7). contracts should be deployed with the same compiler version and flags that they have been tested the most with. Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, the latest compiler which may have higher risks of undiscovered bugs. Contracts may also be deployed by others and the pragma indicates the compiler version intended by the original authors.

Remediation

Remove the ^ sign to lock the pragma version.

Status: Closed. Fixed in version 2.

#Owner privileges (In the period when the owner isn't renounced)

Description
The owner can change the Price at any time.
The owner can open or close the sale.

```
function setMetaNFTPrice(uint256 price) public onlyOwner{
       PRICE_PER_NFT_Meta =price;
    }
    function setSaleIsActive(bool active) external onlyOwner {
        _saleIsActive = active;
    }
```

Remediation
Make these functions internal in next version or the team should announce the investors before open, close the sale or changing the price of the mint to give them time if they want to do anything.
P.S: This issue is common to the majority of NFT smart contracts.

Status: Acknowledged.

Very Low:

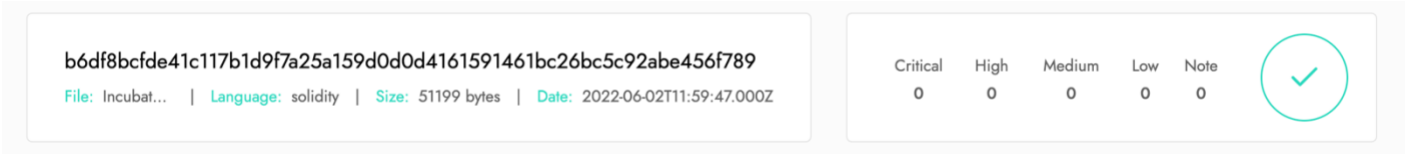
No Very Low severity vulnerabilities were found.

Notes:

No Notes were found.

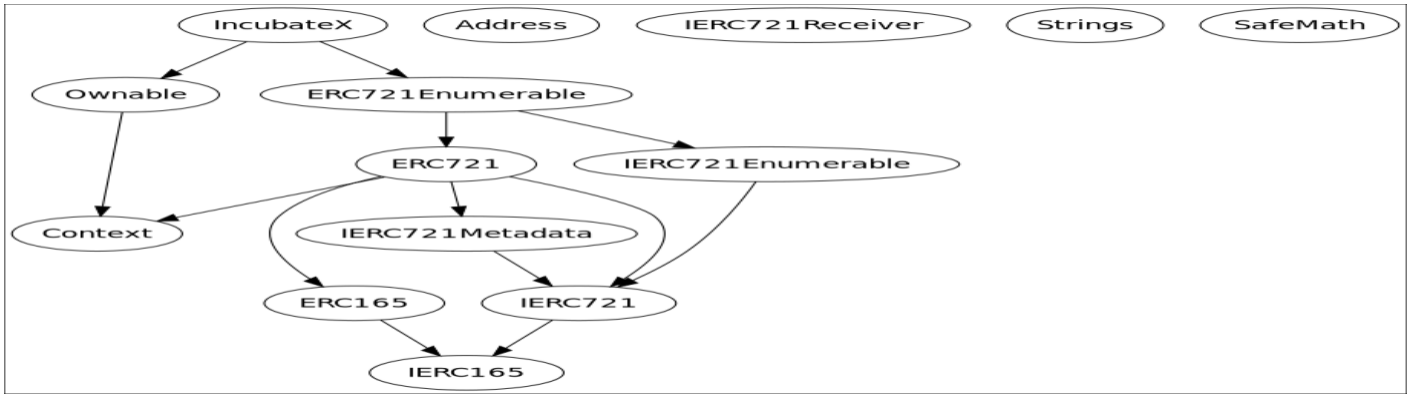
Automatic Testing

1- Check for security

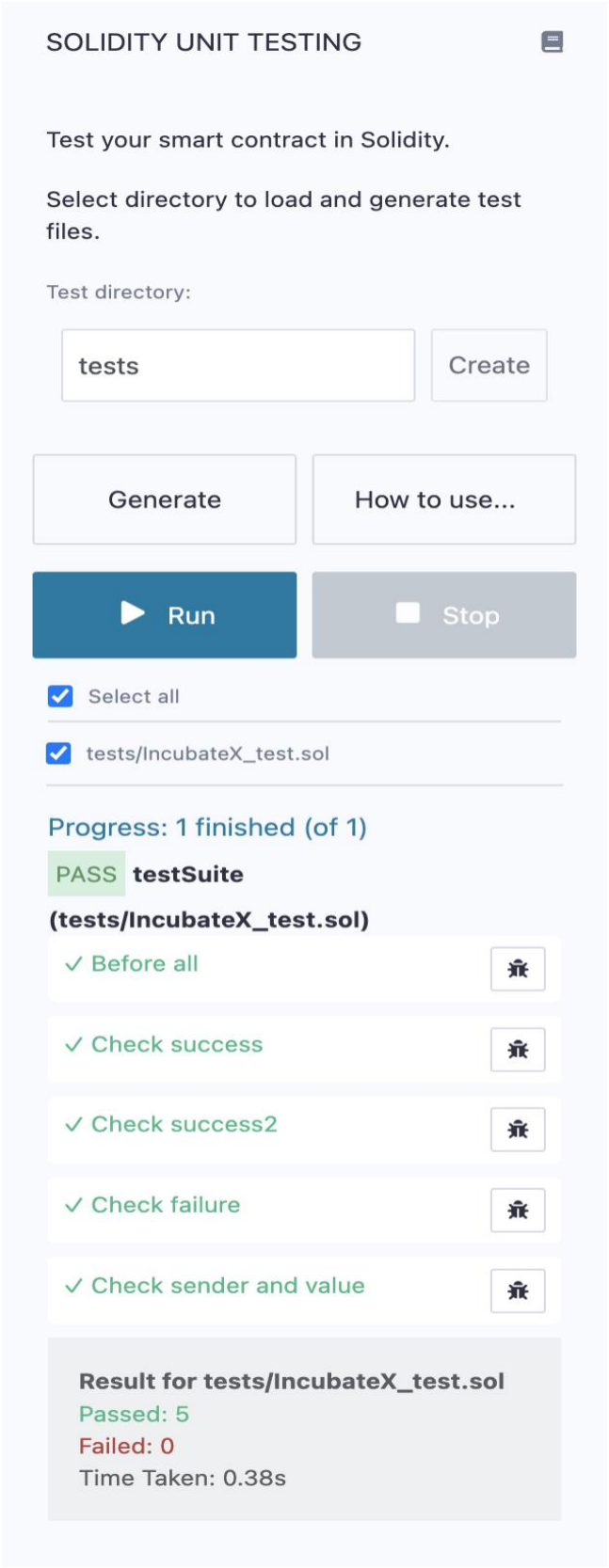


2- SOLIDITY STATIC ANALYSIS

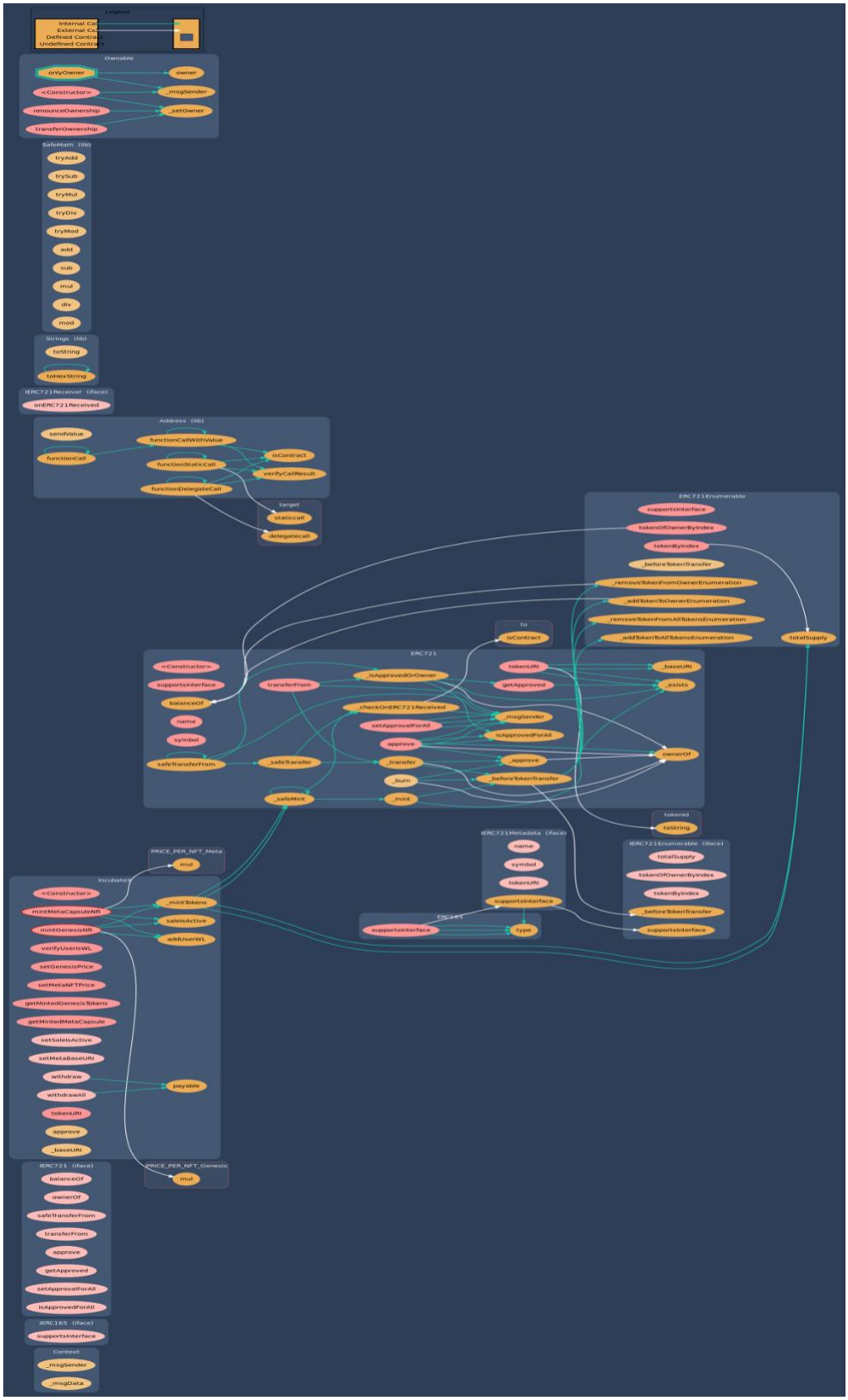
3- Inheritance graph



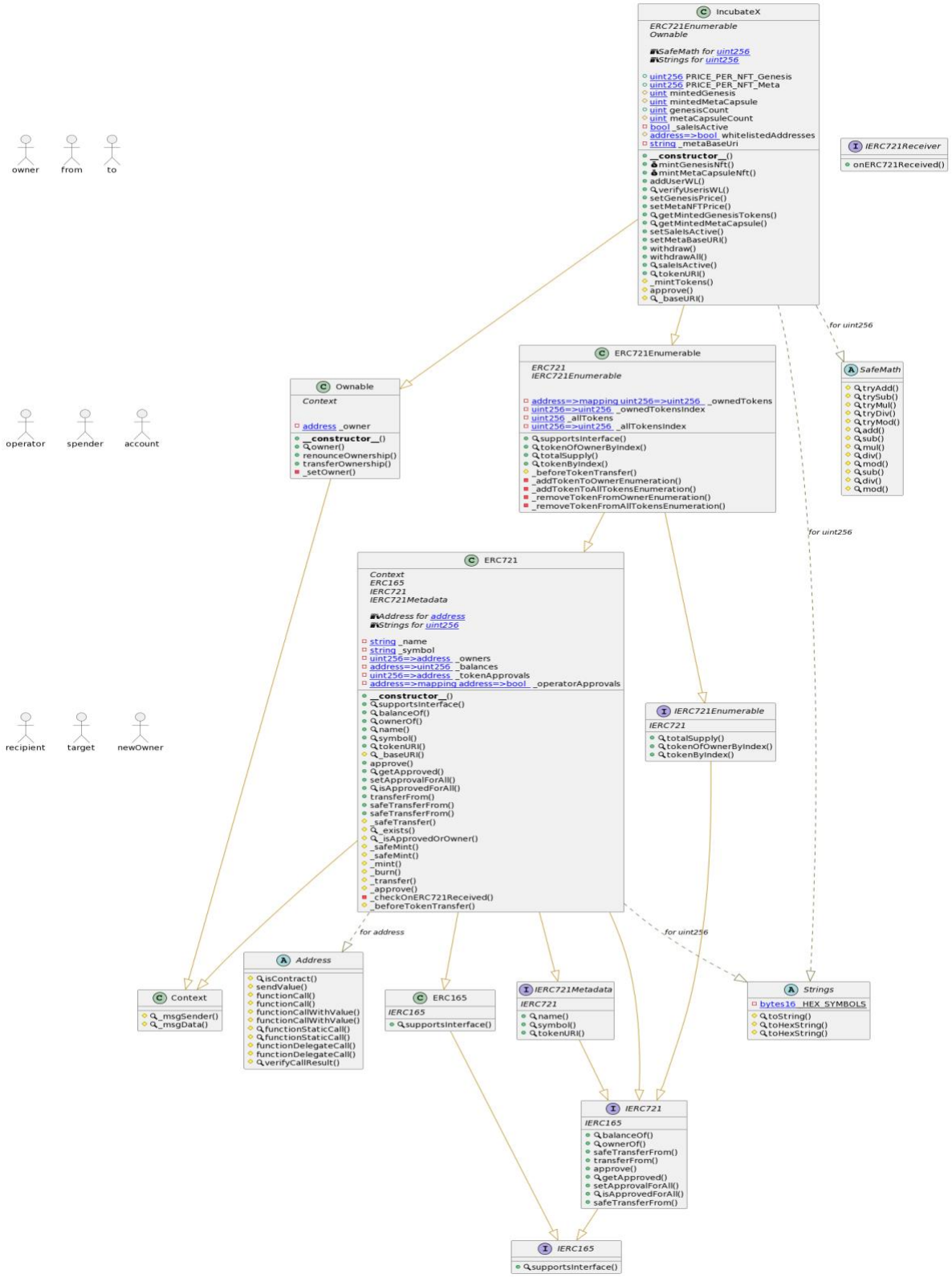
4- SOLIDITY UNIT TESTING



5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Ownable
onlyOwner
owner
<Constructor>
_msgSender
renounceOwnership
_setOwner
transferOwnership
SafeMath (lib)
tryAdd
trySub
tryMul
tryDiv
tryMod
add
sub
mul
div
mod
Strings (lib)
toString
toHexString
IERC721Receiver (iface)
onERC721Received
Address (lib)
sendValue
functionCallWithValue
functionCall
isContract
functionStaticCall
verifyCallResult
functionDelegateCall
target
staticcall
delegatecall
ERC721Enumerable
supportsInterface
tokenOfOwnerByIndex
tokenByIndex
_beforeTokenTransfer
_removeTokenFromOwnerEnumeration
_addTokenToOwnerEnumeration
_removeTokenFromAllTokensEnumeration
_addTokenToAllTokensEnumeration
totalSupply
to
isContract
ERC721
<Constructor>
supportsInterface
balanceOf
name
symbol
safeTransferFrom
transferFrom
_safeTransfer
_isApprovedOrOwner
_checkOnERC721Received
setApprovalForAll
approve
_transfer
_burn
_safeMint
_mint
tokenURI
getApproved
_msgSender
isApprovedForAll
_approve
_beforeTokenTransfer
_baseURI
_exists
ownerOf
tokenId
toString
PRICE_PER_NFT_Meta
mul
Incubate
<Constructor>
mintMetaCapsuleNft
mintGenesisNft
verifyUsersWL
setGenesisPrice
setMetaNFTPrice
getMintedGenesisTokens
getMintedMetaCapsule
setSalesIsActive
setMetaBaseURI
withdraw
withdrawAll
tokenURI
approve
_baseURI
_mintTokens
salesIsActive
addUserWL
payable
IERC721Metadata (iface)
name
symbol
tokenURI
supportsInterface
IERC721Enumerable (iface)
totalSupply
tokenOfOwnerByIndex
tokenByIndex
_beforeTokenTransfer
supportsInterface
IERC165
supportsInterface
type
PRICE_PER_NFT_Genesis
mul
IERC721 (iface)
balanceOf
ownerOf
safeTransferFrom
transferFrom
approve
getApproved
setApprovalForAll
isApprovedForAll
IERC165 (iface)
supportsInterface
Context
_msgSender
_msgData

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
16279055  =>  isContract(address)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
01ffc9a7  =>  supportsInterface(bytes4)
70a08231  =>  balanceOf(address)
6352211e  =>  ownerOf(uint256)
42842e0e  =>  safeTransferFrom(address,address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
095ea7b3  =>  approve(address,uint256)
081812fc  =>  getApproved(uint256)
a22cb465  =>  setApprovalForAll(address,bool)
e985e9c5  =>  isApprovedForAll(address,address)
b88d4fde  =>  safeTransferFrom(address,address,uint256,bytes)
06fdde03  =>  name()
95d89b41  =>  symbol()
c87b56dd  =>  tokenURI(uint256)
743976a0  =>  _baseURI()
24b6b8c0  =>  _safeTransfer(address,address,uint256,bytes)
f8e76cc0  =>  _exists(uint256)
4cdc9549  =>  _isApprovedOrOwner(address,uint256)
b3e1c718  =>  _safeMint(address,uint256)
6a4f832b  =>  _safeMint(address,uint256,bytes)
4e6ec247  =>  _mint(address,uint256)
9b1f9e74  =>  _burn(uint256)
30e0789e  =>  _transfer(address,address,uint256)
7b7d7225  =>  _approve(address,uint256)
1fd01de1  =>  _checkOnERC721Received(address,address,uint256,bytes)
cad3be83  =>  _beforeTokenTransfer(address,address,uint256)
24a084df  =>  sendValue(address,uint256)
a0b5ffb0  =>  functionCall(address,bytes)
241b5886  =>  functionCall(address,bytes,string)
2a011594  =>  functionCallWithValue(address,bytes,uint256)
d525ab8a  =>  functionCallWithValue(address,bytes,uint256,string)
c21d36f3  =>  functionStaticCall(address,bytes)
dbc40fb9  =>  functionStaticCall(address,bytes,string)
ee33b7e2  =>  functionDelegateCall(address,bytes)
57387df0  =>  functionDelegateCall(address,bytes,string)
946b5793  =>  verifyCallResult(bool,bytes,string)
150b7a02  =>  onERC721Received(address,address,uint256,bytes)
18160ddd  =>  totalSupply()
2f745c59  =>  tokenOfOwnerByIndex(address,uint256)
4f6ccce7  =>  tokenByIndex(uint256)
6900a3ae  =>  toString(uint256)
8fba8d5c  =>  toHexString(uint256)
63e1cbea  =>  toHexString(uint256,uint256)
884557bf  =>  tryAdd(uint256,uint256)
a29962b1  =>  trySub(uint256,uint256)
6281efa4  =>  tryMul(uint256,uint256)
736ecb18  =>  tryDiv(uint256,uint256)
38dc0867  =>  tryMod(uint256,uint256)
771602f7  =>  add(uint256,uint256)
```

```
b67d77c5  =>  sub(uint256,uint256)
c8a4ac9c  =>  mul(uint256,uint256)
a391c15b  =>  div(uint256,uint256)
f43f523a  =>  mod(uint256,uint256)
e31bdc0a  =>  sub(uint256,uint256,string)
b745d336  =>  div(uint256,uint256,string)
71af23e8  =>  mod(uint256,uint256,string)
8da5cb5b  =>  owner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
fc201122  =>  _setOwner(address)
69025b5f  =>  _addTokenToOwnerEnumeration(address,uint256)
e03d890b  =>  _addTokenToAllTokensEnumeration(uint256)
68df0d53  =>  _removeTokenFromOwnerEnumeration(address,uint256)
4cbb4a0a  =>  _removeTokenFromAllTokensEnumeration(uint256)
c16fe148  =>  mintGenesisNft(uint16)
b49cc889  =>  mintMetaCapsuleNft(uint16)
d50a165b  =>  addUserWL(address)
1c5c4200  =>  verifyUserisWL(address)
61f36462  =>  setGenesisPrice(uint256)
9b8b4533  =>  setMetaNFTPrice(uint256)
5dc2fefc  =>  getMintedGenesisTokens()
c8050c71  =>  getMintedMetaCapsule()
02c88989  =>  setSaleIsActive(bool)
18d272eb  =>  setMetaBaseURI(string)
2e1a7d4d  =>  withdraw(uint256)
853828b6  =>  withdrawAll()
eb8d2444  =>  saleIsActive()
89c590ab  =>  _mintTokens(uint16,address,uint256)
febe4909  =>  approve(uint256,address)
```

Automatic general report

```
Files Description Table

|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/IncubateX.sol |
168e2cb797fa871a0662e29504bc145f960cf2db |

 Contracts Description Table

|  Contract  |         Type        |       Bases      |                  |
|
|:----------:|:-------------------:|:----------------:|:----------------:|:--------
-------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |
**Modifiers**  |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | Internal 🔒 |   | |
||||||
| **IERC165** | Interface |  |||
| └ | supportsInterface | External ❗ |   |NO❗ |
||||||
| **IERC721** | Interface | IERC165 |||
| └ | balanceOf | External ❗ |   |NO❗ |
| └ | ownerOf | External ❗ |   |NO❗ |
| └ | safeTransferFrom | External ❗ | 🛑 |NO❗ |
| └ | transferFrom | External ❗ | 🛑 |NO❗ |
| └ | approve | External ❗ | 🛑 |NO❗ |
| └ | getApproved | External ❗ |   |NO❗ |
| └ | setApprovalForAll | External ❗ | 🛑 |NO❗ |
| └ | isApprovedForAll | External ❗ |   |NO❗ |
| └ | safeTransferFrom | External ❗ | 🛑 |NO❗ |
||||||
| **IERC721Metadata** | Interface | IERC721 |||
| └ | name | External ❗ |   |NO❗ |
| └ | symbol | External ❗ |   |NO❗ |
| └ | tokenURI | External ❗ |   |NO❗ |
||||||
| **ERC165** | Implementation | IERC165 |||
| └ | supportsInterface | Public ❗ |   |NO❗ |
||||||
| **ERC721** | Implementation | Context, ERC165, IERC721, IERC721Metadata |||
| └ | <Constructor> | Public ❗ | 🛑 |NO❗ |
| └ | supportsInterface | Public ❗ |   |NO❗ |
| └ | balanceOf | Public ❗ |   |NO❗ |
| └ | ownerOf | Public ❗ |   |NO❗ |
| └ | name | Public ❗ |   |NO❗ |
| └ | symbol | Public ❗ |   |NO❗ |
| └ | tokenURI | Public ❗ |   |NO❗ |
| └ | _baseURI | Internal 🔒 |   | |
| └ | approve | Public ❗ | 🛑 |NO❗ |
```

└	getApproved	Public ❗		NO❗
└	setApprovalForAll	Public ❗	🛑	NO❗
└	isApprovedForAll	Public ❗		NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	safeTransferFrom	Public ❗	🛑	NO❗
└	safeTransferFrom	Public ❗	🛑	NO❗
└	_safeTransfer	Internal 🔒	🛑	
└	_exists	Internal 🔒		
└	_isApprovedOrOwner	Internal 🔒		
└	_safeMint	Internal 🔒	🛑	
└	_safeMint	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_transfer	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_checkOnERC721Received	Private 🔐	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResult	Internal 🔒		
IERC721Receiver	Interface			
└	onERC721Received	External ❗	🛑	NO❗
IERC721Enumerable	Interface	IERC721		
└	totalSupply	External ❗		NO❗
└	tokenOfOwnerByIndex	External ❗		NO❗
└	tokenByIndex	External ❗		NO❗
Strings	Library			
└	toString	Internal 🔒		
└	toHexString	Internal 🔒		
└	toHexString	Internal 🔒		
SafeMath	Library			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		

└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
ERC721Enumerable	Implementation	ERC721, IERC721Enumerable		
└	supportsInterface	Public ❗		NO❗
└	tokenOfOwnerByIndex	Public ❗		NO❗
└	totalSupply	Public ❗		NO❗
└	tokenByIndex	Public ❗		NO❗
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_addTokenToOwnerEnumeration	Private 🔐	🛑	
└	_addTokenToAllTokensEnumeration	Private 🔐	🛑	
└	_removeTokenFromOwnerEnumeration	Private 🔐	🛑	
└	_removeTokenFromAllTokensEnumeration	Private 🔐	🛑	
IncubateX	Implementation	ERC721Enumerable, Ownable		
└	<Constructor>	Public ❗	🛑	ERC721
└	mintGenesisNft	Public ❗	💵	NO❗
└	mintMetaCapsuleNft	Public ❗	💵	NO❗
└	addUserWL	Public ❗	🛑	NO❗
└	verifyUserisWL	Public ❗		NO❗
└	setGenesisPrice	Public ❗	🛑	onlyOwner
└	setMetaNFTPrice	Public ❗	🛑	onlyOwner
└	getMintedGenesisTokens	Public ❗		NO❗
└	getMintedMetaCapsule	Public ❗		NO❗
└	setSaleIsActive	External ❗	🛑	onlyOwner
└	setMetaBaseURI	External ❗	🛑	onlyOwner
└	withdraw	External ❗	🛑	onlyOwner
└	withdrawAll	External ❗	🛑	onlyOwner
└	saleIsActive	Public ❗		NO❗
└	tokenURI	Public ❗		NO❗
└	_mintTokens	Internal 🔒	🛑	
└	approve	Internal 🔒	🛑	
└	_baseURI	Internal 🔒		

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues in version 2. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “ Secured”.

✓ No Critical logic issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of 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 the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below 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 Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico 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 Saferico 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, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, 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 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.