

Smart Contract Security Audit V1

Up World Smart Contract

20/6/2022



<https://saferico.com/>

business@saferico.com

https://t.me/SFI_ANN

—

Table of Contents

Table of Contents

Background

Project Information

NFT Information

Executive Summary

File and Function Level Report

File in Scope:

Issues Checking Status

Severity Definitions

Audit Findings

Automatic testing

Testing proves

Inheritance graph

Call graph

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

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:** Ethereum
- **Contract Address:** 0x7481cd6984e02cd6d68917da6c49c5163dfecfd6
- **Code:**

<https://github.com/Saferico/Smart-Contracts-for-Projects/blob/main/UP%20World.sol>

NFT Information

- **Name:** UP
- **MAX Supply:** Max Supply1 tier = 2500, Max Supply2 tier = 1000, Max Supply 3tier = 750, Max Supply4 tier = 400, and Max Supply5 tier = 150.

Contracts address deployed to test net (Ethereum)

Up World smart contract on ETH test net to test every function by the auditor.

<https://rinkeby.etherscan.io/address/0x7481cd6984e02cd6d68917da6c49c5163dfecfd6>

Executive Summary

According to our assessment, the customer`s solidity smart contract is **“WELL SECURED”**. The team has fixed the low-level issues.

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, 1 medium, 3 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

UpWorld.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
UpWorld.sol	856fc9a7b2eb45fc220aa7c6a7fe24e886d82c2483b7c3a6cf5718ad90b59b27	0x7481cd6984e02cd6d68917da6c49c5163dfecfd6

- Contract: UpWorld
- Inherit: ERC721AQueryable, 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
Costttier1	✓	Read / public	Passed
supportsInterface	✓	Read / public	Passed
Costttier2	✓	Read / public	Passed
balanceOf	✓	Read / public	Passed
Owner	✓	Read / public	Passed
Costttier3	✓	Read / public	Passed
tokenOfOwner	✓	Read / public	Passed
getApprovedForAll	✓	Read / public	Passed
NFTTier	✓	Read / public	Passed
getApproved	✓	Read / public	Passed

ownerOf	✓	Read / public	Passed
tokenURI	✓	Read / public	Passed
totalSupply	✓	Read / public	Passed
Costtier4	✓	Read / public	Passed
Costtier5	✓	Read / public	Passed
counter1	✓	Read / public	Passed
counter2	✓	Read / public	Passed
counter3	✓	Read / public	Passed
counter4	✓	Read / public	Passed
counter5	✓	Read / public	Passed
freeze	✓	Read / public	Passed
maxSupply1tier	✓	Read / public	Passed
maxSupply2tier	✓	Read / public	Passed
maxSupply3tier	✓	Read / public	Passed
maxSupply4tier	✓	Read / public	Passed
maxSupply5tier	✓	Read / public	Passed
maxMintAmountPerWallet	✓	Read / public	Passed
nftCharacterAddress	✓	Read / public	Passed
NFTPerAddressforTier1	✓	Read / public	Passed
NFTPerAddressforTier2	✓	Read / public	Passed
NFTPerAddressforTier3	✓	Read / public	Passed
NFTPerAddressforTier4	✓	Read / public	Passed
NFTPerAddressforTier5	✓	Read / public	Passed
paused	✓	Read / public	Passed
SpecialNftAddress	✓	Read / public	Passed
reserve	✓	Read / public	Passed
approve	✓	Write / public	Passed

MintTier1	✓	Write / payable	Passed
MintTier2	✓	Write / payable	Passed
MintTier3	✓	Write / payable	Passed
MintTier4	✓	Write / payable	Passed
MintTier5	✓	Write / payable	Passed
Reserve	✓	Write / public	Passed
transferOwnership	✓	Write / public	Passed
setApprovalForAll	✓	Write / public	Passed
transferFrom	✓	Write / public	Passed
withdraw	✓	Write / public	Passed
safeTransferFrom	✓	Write / public	Passed
renounceOwnership	✓	Write / public	Passed
safeTransferFrom	✓	Write / public	Passed
setCost1	✓	Write / public	Passed
setCost2	✓	Write / public	Passed
setCost3	✓	Write / public	Passed
setCost4	✓	Write / public	Passed
setCost5	✓	Write / public	Passed
setFreeze	✓	Write / public	Passed
setMaxSupplyforTier1	✓	Write / public	Passed
setMaxSupplyforTier2	✓	Write / public	Passed
setMaxSupplyforTier3	✓	Write / public	Passed
setMaxSupplyforTier4	✓	Write / public	Passed
setMaxSupplyforTier5	✓	Write / public	Passed
setPaused	✓	Write / public	Passed
setSpecialNftAddress	✓	Write / public	Passed
setMaxTxPerAddress	✓	Write / public	Passed

setNftCharacterAddress	✓	Write / public	Passed
setUriPrefix	✓	Write / public	Passed
burnAndReveal	✓	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 with Notes
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed with Notes
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:

No Critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found.

Medium:

#Centralization Risks

Description

The owner has the authority to :

- Freeze and unfreeze NFTs for trading and transfer from an address to any address.
- Pause and unpause the mint of the NFTs.
- Change the price, max supply, and the max amount of NFT per wallet.

```
function _beforeTokenTransfers(address from, address to, uint256 tokenId , uint
quantity)
    internal
    override
    {

        require(freeze == false , "Eggs are frozen");
        quantity=1;
        super._beforeTokenTransfers(from, to, tokenId , quantity);
    }

    function setMaxSupplyforTier1(uint16 _maxSupply) external onlyOwner {
        maxSupply1tier = _maxSupply;
    }
    function setMaxSupplyforTier2(uint16 _maxSupply) external onlyOwner {
        maxSupply2tier = _maxSupply;
    }
    function setMaxSupplyforTier3(uint16 _maxSupply) external onlyOwner {
        maxSupply3tier = _maxSupply;
    }
    function setMaxSupplyforTier4(uint16 _maxSupply) external onlyOwner {
        maxSupply4tier = _maxSupply;
    }
    function setMaxSupplyforTier5(uint16 _maxSupply) external onlyOwner {
        maxSupply5tier = _maxSupply;
    }

    function setFreeze() external onlyOwner {
```

```

        freeze = !freeze;
    }
    function setPaused() external onlyOwner {
        paused = !paused;
    }
    function setCost1(uint256 _cost) external onlyOwner {
        Costtttier1 = _cost;
        delete _cost;
    }
    function setCost2(uint256 _cost) external onlyOwner {
        Costtttier2 = _cost;
        delete _cost;
    }
    function setCost3(uint256 _cost) external onlyOwner {
        Costtttier3 = _cost;
        delete _cost;
    }
    function setCost4(uint256 _cost) external onlyOwner {
        Costtttier4 = _cost;
        delete _cost;
    }
    function setCost5(uint256 _cost) external onlyOwner {
        Costtttier5 = _cost;
        delete _cost;
    }
}

function setMaxTxPerAddress(uint8 _limit) external onlyOwner{
    maxMintAmountPerWallet = _limit;
    delete _limit;
}

```

Remediation

Make these functions internal in next version or the team should announce the investors before change anything to give them time if they want to do anything.

P.S: This issue is common to the majority of NFT's smart contracts.

Status: [Acknowledged by the Auditee](#)

Low:

#Missing zero address validation

Description

When the owner wants to reserve some NFTs for the investors or add an NFT character address or special NFT address, he has to check for the zero address to make, he didn't add the burn address. Otherwise, the reserve will act like a burn function and the other functions will not work correctly.

```

function Reserve(address _receiver , uint8 marker ,uint8 _mintAmount) external
onlyOwner {
    uint16 totalSupply = uint16(totalSupply());
    _safeMint(_receiver , _mintAmount);
    setTierMarker(totalSupply , _mintAmount , marker);

    delete _mintAmount;
    delete _receiver;
    delete totalSupply;
}
function setNftCharacterAddress(address _address) external onlyOwner {
    nftCharacterAddress = _address;
    nftCharacter = INftCharacter(nftCharacterAddress);
}
function setSpecialNftAddress(address _address) external onlyOwner {
    SpecialNftAddress = _address;
    specialNft = ISpecialNft(SpecialNftAddress);
}

```

Remediation

Use the require statement to check for zero addresses.

Status: **Closed**. Fixed in version 2.

#Multiple pragma statements

Line	Pragma
5	pragma solidity ^0.8.0;
75	pragma solidity ^0.8.0;
102	pragma solidity ^0.8.1;
327	pragma solidity ^0.8.0;
357	pragma solidity ^0.8.0;
385	pragma solidity ^0.8.1;
416	pragma solidity ^0.8.0;
561	pragma solidity ^0.8.0;
590	pragma solidity ^0.8.4;
1214	pragma solidity ^0.8.0;
1370	pragma solidity ^0.8.7;

Description

There are multiple pragma statements in the code. The newest compiler version 0.8.14 will work with the code, but keeping only one pragma statement helps in maintaining readability of the code.

Remediation

Keep a single pragma statement.

Status: **Closed**. Fixed In version 2

#Use of block.timestamp for comparisons

Description

The value of block.timestamp can be manipulated by the miner.
And conditions with strict equality is difficult to achieve -
block.timestamp

Remediation

Avoid use of block.timestamp

Status: **Acknowledged**

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Missing SPDX-License-Identifier:

Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: UNLICENSED" for non-open-source code. Please see <https://spdx.org> for more information .

Remediation

Add License Identifier

// SPDX-License-Identifier: MIT

Status: **Closed**. Fixed In version 2

Automatic Testing

1- Check for security

856fc9a7b2eb45fc220aa7c6a7fe24e886d82c2483b7c3a6cf5718ad90b59b27
File: UP Wor... | Language: solidity | Size: 57092 bytes | Date: 2022-06-20T03:51:48.718Z

Critical	High	Medium	Low	Note
0	0	0	0	0



2- SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

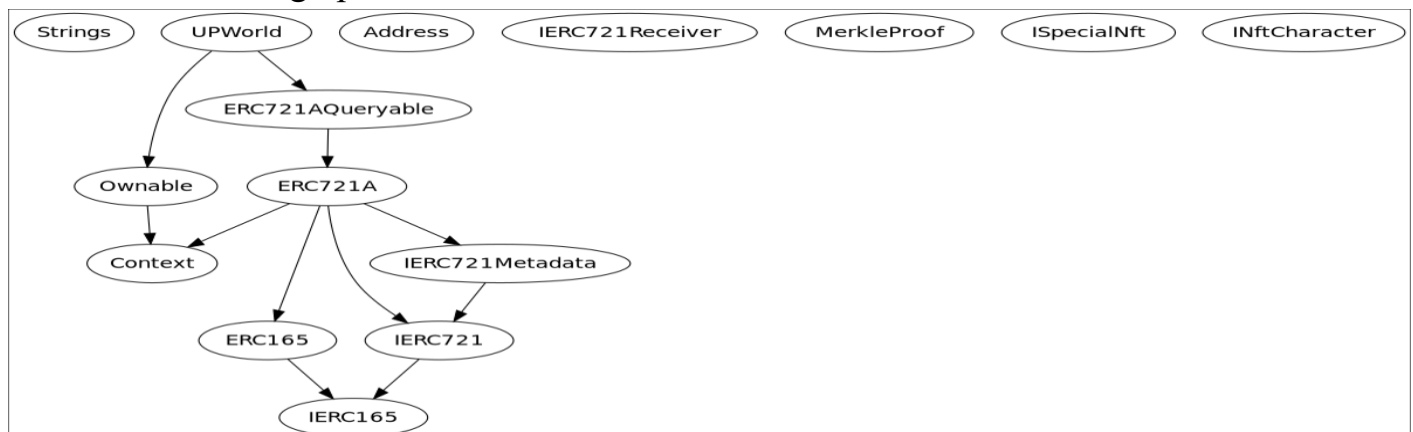
☒ Select all ☒ Autorun Run

- Security**
 - ☒ Select Security
 - ☒ **Transaction origin:**
'tx.origin' used
 - ☒ **Check-effects-interaction:**
Potential reentrancy bugs
 - ☒ **Inline assembly:**
Inline assembly used
 - ☒ **Block timestamp:**
Can be influenced by miners
 - ☒ **Low level calls:**
Should only be used by experienced devs
 - ☒ **Block hash:**
Can be influenced by miners
 - ☒ **Selfdestruct:**
Contracts using destructed contract can be broken
- Gas & Economy**
 - ☒ Select Gas & Economy
 - ☒ **Gas costs:**
Too high gas requirement of functions
 - ☒ **This on local calls:**
Invocation of local functions via 'this'
 - ☒ **Delete dynamic array:**
Use require/assert to ensure complete deletion
 - ☒ **For loop over dynamic array:**
Iterations depend on dynamic array's size
 - ☒ **Ether transfer in loop:**
Transferring Ether in a for/while/do-while loop

SOLIDITY STATIC ANALYSIS

- ERC**
 - ☒ Select ERC
 - ☒ **ERC20:**
'decimals' should be 'uint8'
- Miscellaneous**
 - ☒ Select Miscellaneous
 - ☒ **Constant/View/Pure functions:**
Potentially constant/view/pure functions
 - ☒ **Similar variable names:**
Variable names are too similar
 - ☒ **No return:**
Function with 'returns' not returning
 - ☒ **Guard conditions:**
Ensure appropriate use of require/assert
 - ☒ **Result not used:**
The result of an operation not used
 - ☒ **String length:**
Bytes length != String length
 - ☒ **Delete from dynamic array:**
'delete' leaves a gap in array
 - ☒ **Data truncated:**
Division on int/uint values truncates the result

3- Inheritance graph



4- SOLIDITY UNIT TESTING

SOLIDITY UNIT TESTING

Test your smart contract in Solidity.

Select directory to load and generate test files.

Test directory:

☒ Select all

☒ tests/UP World_test.sol

Progress: 1 finished (of 1)

PASS testSuite (tests/UP World_test.sol)

✓ Before all

✓ Check success

✓ Check success2

✓ Check failure

✓ Check sender and value

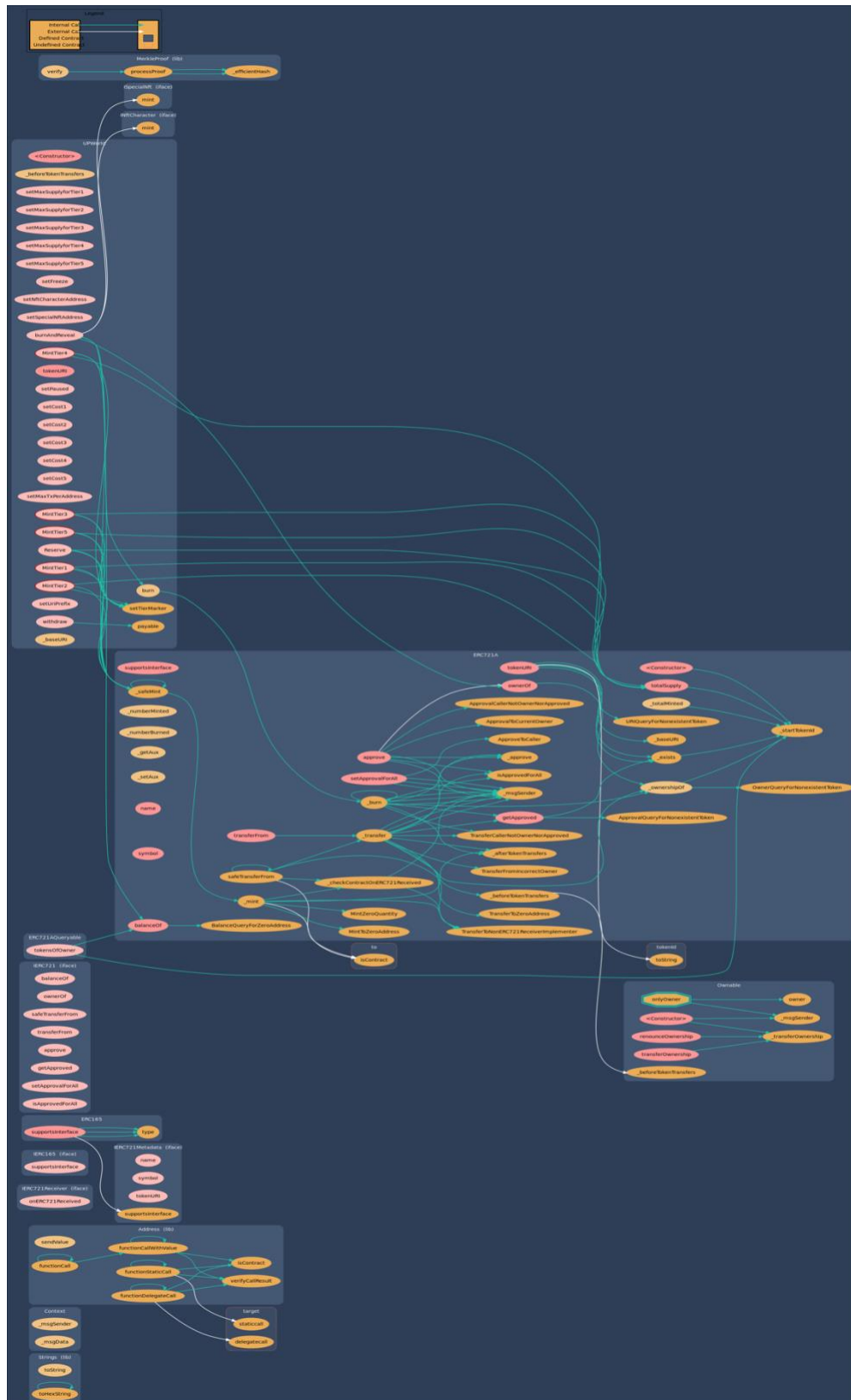
Result for tests/UP World_test.sol

Passed: 5

Failed: 0

Time Taken: 0.33s

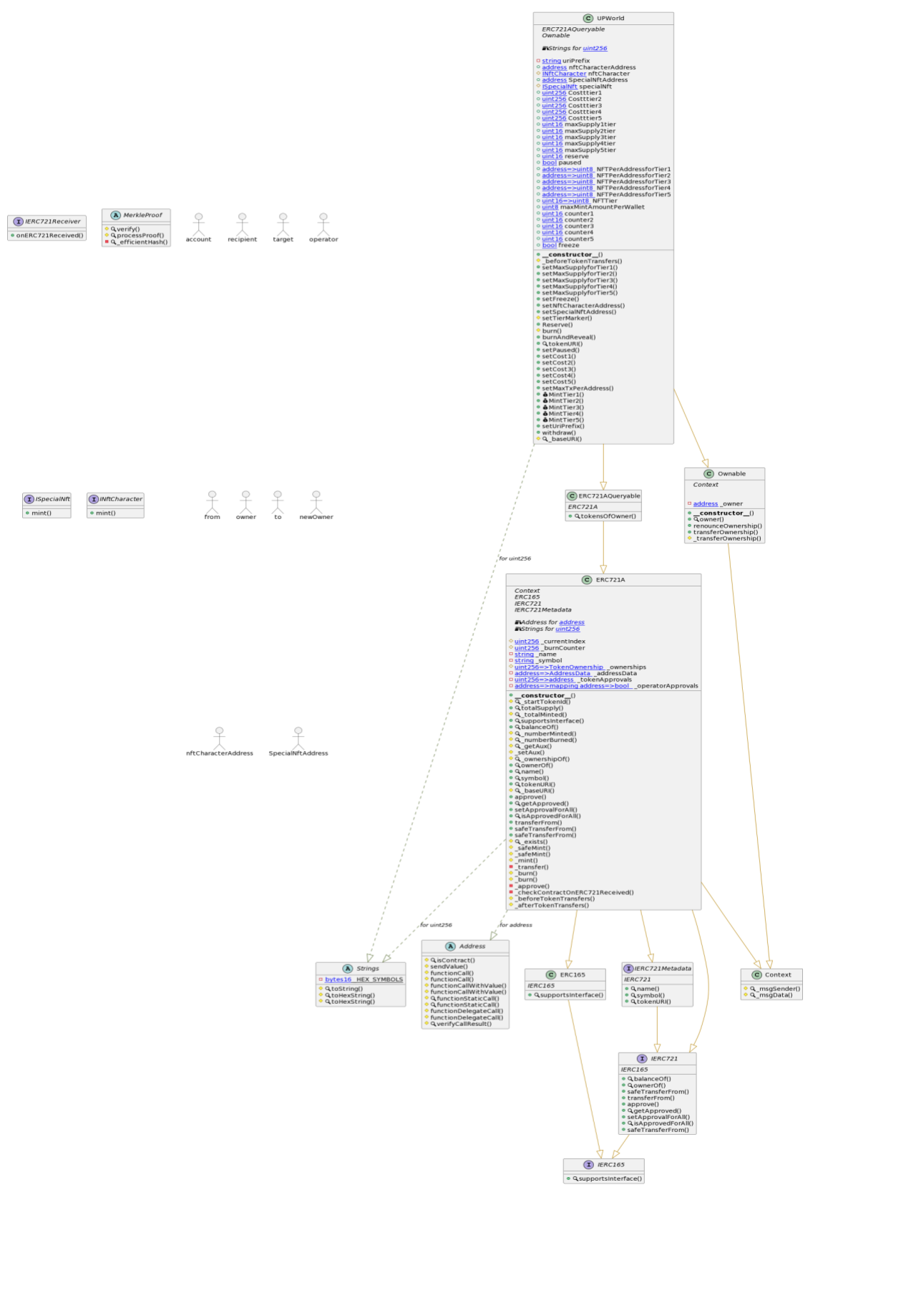
5- Call graph



The diagram illustrates the relationships between various smart contract interfaces and implementations. The main components are:

- ERC721Receiver**: An interface with a method `onERC721Received()`.
- MerkleProof**: An interface with methods `qVerify()`, `qprocessProof()`, and `qEfficientHash()`.
- ERC721Queryable**: An interface with methods `QtokensOfOwner()` and `QbaseURI()`.
- ERC721**: A class implementing the `ERC721Queryable` interface. It has a `Context` and `ERC165` interface. It implements methods like `QstartTokens()`, `QtotalSupply()`, `QtotalMinted()`, `QsupportsInterface()`, `QbalanceOf()`, `QnumberMinted()`, `QgetAux()`, `QownershipOf()`, `QownerOf()`, `Qname()`, `Qsymbol()`, `QtokensURI()`, `QbaseURI()`, `approve()`, `QgetApproved()`, `setApprovalForAll()`, `QisApprovedForAll()`, `transferFrom()`, `safeTransferFrom()`, `Qexists()`, `safeMint()`, `mint()`, `transfer()`, `burn()`, `checkContractOnERC721Received()`, `beforeTokenTransfers()`, and `afterTokenTransfers()`.
- ERC165**: An interface with a method `QsupportsInterface()`.
- ERC721Metadata**: An interface with methods `Qname()`, `Qsymbol()`, and `QtokensURI()`.
- Context**: An interface with methods `QmsgSender()` and `QmsgData()`.
- Strings**: A class implementing the `Strings` interface. It has a `Context` and `ERC165` interface. It implements methods like `QstartTokens()`, `QtotalSupply()`, `QtotalMinted()`, `QsupportsInterface()`, `QbalanceOf()`, `QnumberMinted()`, `QgetAux()`, `QownershipOf()`, `QownerOf()`, `Qname()`, `Qsymbol()`, `QtokensURI()`, `QbaseURI()`, `approve()`, `QgetApproved()`, `setApprovalForAll()`, `QisApprovedForAll()`, `transferFrom()`, `safeTransferFrom()`, `Qexists()`, `safeMint()`, `mint()`, `transfer()`, `burn()`, `checkContractOnERC721Received()`, `beforeTokenTransfers()`, and `afterTokenTransfers()`.
- Address**: A class implementing the `Address` interface. It has a `Context` and `ERC165` interface. It implements methods like `QstartTokens()`, `QtotalSupply()`, `QtotalMinted()`, `QsupportsInterface()`, `QbalanceOf()`, `QnumberMinted()`, `QgetAux()`, `QownershipOf()`, `QownerOf()`, `Qname()`, `Qsymbol()`, `QtokensURI()`, `QbaseURI()`, `approve()`, `QgetApproved()`, `setApprovalForAll()`, `QisApprovedForAll()`, `transferFrom()`, `safeTransferFrom()`, `Qexists()`, `safeMint()`, `mint()`, `transfer()`, `burn()`, `checkContractOnERC721Received()`, `beforeTokenTransfers()`, and `afterTokenTransfers()`.
- ERC165**: A class implementing the `ERC165` interface. It has a `Context` and `ERC165` interface. It implements methods like `QstartTokens()`, `QtotalSupply()`, `QtotalMinted()`, `QsupportsInterface()`, `QbalanceOf()`, `QnumberMinted()`, `QgetAux()`, `QownershipOf()`, `QownerOf()`, `Qname()`, `Qsymbol()`, `QtokensURI()`, `QbaseURI()`, `approve()`, `QgetApproved()`, `setApprovalForAll()`, `QisApprovedForAll()`, `transferFrom()`, `safeTransferFrom()`, `Qexists()`, `safeMint()`, `mint()`, `transfer()`, `burn()`, `checkContractOnERC721Received()`, `beforeTokenTransfers()`, and `afterTokenTransfers()`.

The diagram also shows actors (account, recipient, target, operator, from, owner, to, newOwner) and their interactions with the contracts. The diagram is organized into several sections, with a large central section for ERC721 and related classes, and smaller sections for ERC721Receiver, MerkleProof, and ERC165. Arrows indicate dependencies and associations between the classes and actors.



Functions signature

Sighash		Function Signature
=====		
16279055	=>	isContract (address)
6900a3ae	=>	toString (uint256)
8fba8d5c	=>	toHexString (uint256)
63e1cbea	=>	toHexString (uint256, uint256)
119df25f	=>	_msgSender ()
8b49d47e	=>	_msgData ()
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)
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)
98995f77	=>	_startTokenId ()
18160ddd	=>	totalSupply ()
736bf591	=>	_totalMinted ()
4d388a98	=>	_numberMinted (address)
6ba1b8d0	=>	_numberBurned (address)
f4a540c5	=>	_getAux (address)
4ff8c452	=>	_setAux (address, uint64)
fb372cf2	=>	_ownershipOf (uint256)
743976a0	=>	_baseURI ()
f8e76cc0	=>	_exists (uint256)
b3e1c718	=>	_safeMint (address, uint256)
6a4f832b	=>	_safeMint (address, uint256, bytes)
de0d9900	=>	_mint (address, uint256, bytes, bool)
30e0789e	=>	_transfer (address, address, uint256)
9b1f9e74	=>	_burn (uint256)
834a9477	=>	_burn (uint256, bool)
f272404d	=>	_approve (address, uint256, address)
d88343e2	=>	_checkContractOnERC721Received (address, address, uint256, bytes)
ef435773	=>	_beforeTokenTransfers (address, address, uint256, uint256)
08c018f7	=>	_afterTokenTransfers (address, address, uint256, uint256)
5a9a49c7	=>	verify (bytes32 [], bytes32, bytes32)

```
62702a6b => processProof(bytes32[],bytes32)
41ed615b => _efficientHash(bytes32,bytes32)
8da5cb5b => owner()
715018a6 => renounceOwnership()
f2fde38b => transferOwnership(address)
d29d44ee => _transferOwnership(address)
8462151c => tokensOfOwner(address)
691562a0 => mint(address,uint8)
d3ff2e8f => setMaxSupplyforTier1(uint16)
6332a87b => setMaxSupplyforTier2(uint16)
83413c81 => setMaxSupplyforTier3(uint16)
e7b2019f => setMaxSupplyforTier4(uint16)
623ad2d8 => setMaxSupplyforTier5(uint16)
2c8cbe40 => setFreeze()
d686aa7b => setNftCharacterAddress(address)
a8a5b903 => setSpecialNftAddress(address)
08cc399c => setTierMarker(uint16,uint8,uint8)
6e4483b0 => Reserve(address,uint8,uint8)
42966c68 => burn(uint256)
06489201 => burnAndReveal(uint16)
37a66d85 => setPaused()
b08a55f6 => setCost1(uint256)
1950c2fc => setCost2(uint256)
694aaf5b => setCost3(uint256)
0a4f1c9e => setCost4(uint256)
153fa2e4 => setCost5(uint256)
1d957333 => setMaxTxPerAddress(uint8)
84f9ce9c => MintTier1(uint8)
3fb21249 => MintTier2(uint8)
19cd5e23 => MintTier3(uint8)
e924ef06 => MintTier4(uint8)
2246094f => MintTier5(uint8)
7ec4a659 => setUriPrefix(string)
3ccfd60b => withdraw()
```

Automatic general report

Files Description Table

```
| File Name | SHA-1 Hash |
|-----|-----|
| /Users/macbook/Desktop/smart contracts/UP World.sol |
724bf4815ec320a221dce6628b026c4cea23a919 |
```

Contracts Description Table







































```

Contract | Type | Bases |
:-----:|:-----:|:-----:|:-----:|:-----:|
L |
**Modifiers** |
|||||
**Strings** | Library | |||
L | toString | Internal | 🔒 | | |
L | toHexString | Internal | 🔒 | | |
L | toHexString | Internal | 🔒 | | |
|||||
**Context** | Implementation | |||
L | _msgSender | Internal | 🔒 | | |
L | _msgData | Internal | 🔒 | | |
|||||
**Address** | Library | |||
L | isContract | Internal | 🔒 | | |
L | sendValue | Internal | 🔒 | 🔒 | | |
L | functionCall | Internal | 🔒 | 🔒 | | |
L | functionCall | Internal | 🔒 | 🔒 | | |
L | functionCallWithValue | Internal | 🔒 | 🔒 | | |
L | functionCallWithValue | Internal | 🔒 | 🔒 | | |
L | functionStaticCall | Internal | 🔒 | | |
L | functionStaticCall | Internal | 🔒 | | |
L | functionDelegateCall | Internal | 🔒 | 🔒 | | |
L | functionDelegateCall | Internal | 🔒 | 🔒 | | |
L | verifyCallResult | Internal | 🔒 | | |
|||||
**IERC721Receiver** | Interface | |||
L | onERC721Received | External | ⚠️ | 🔒 | NO⚠️ |
|||||
**IERC165** | Interface | |||
L | supportsInterface | External | ⚠️ | | NO⚠️ |
|||||
**ERC165** | Implementation | IERC165 | |||
L | supportsInterface | Public | ⚠️ | | NO⚠️ |
|||||
**IERC721** | Interface | IERC165 | |||
L | balanceOf | External | ⚠️ | | NO⚠️ |
L | ownerOf | External | ⚠️ | | NO⚠️ |
L | safeTransferFrom | External | ⚠️ | 🔒 | NO⚠️ |
L | transferFrom | External | ⚠️ | 🔒 | NO⚠️ |

```



L	approve	External	!	⊗	NO	!	
L	getApproved	External	!		NO	!	
L	setApprovalForAll	External	!	⊗	NO	!	
L	isApprovedForAll	External	!		NO	!	
L	safeTransferFrom	External	!	⊗	NO	!	
IERC721Metadata		Interface		IERC721			
L	name	External	!		NO	!	
L	symbol	External	!		NO	!	
L	tokenURI	External	!		NO	!	
ERC721A		Implementation		Context, ERC165, IERC721, IERC721Metadata			
L	<Constructor>	Public	!	⊗	NO	!	
L	_startTokenId	Internal		🔒			
L	totalSupply	Public	!		NO	!	
L	_totalMinted	Internal		🔒			
L	supportsInterface	Public	!		NO	!	
L	balanceOf	Public	!		NO	!	
L	_numberMinted	Internal		🔒			
L	_numberBurned	Internal		🔒			
L	_getAux	Internal		🔒			
L	_setAux	Internal		🔒	⊗		
L	_ownershipOf	Internal		🔒			
L	ownerOf	Public	!		NO	!	
L	name	Public	!		NO	!	
L	symbol	Public	!		NO	!	
L	tokenURI	Public	!		NO	!	
L	_baseURI	Internal		🔒			
L	approve	Public	!	⊗	NO	!	
L	getApproved	Public	!		NO	!	
L	setApprovalForAll	Public	!	⊗	NO	!	
L	isApprovedForAll	Public	!		NO	!	
L	transferFrom	Public	!	⊗	NO	!	
L	safeTransferFrom	Public	!	⊗	NO	!	
L	safeTransferFrom	Public	!	⊗	NO	!	
L	_exists	Internal		🔒			
L	_safeMint	Internal		🔒	⊗		
L	_safeMint	Internal		🔒	⊗		
L	_mint	Internal		🔒	⊗		
L	_transfer	Private		🔒	⊗		
L	_burn	Internal		🔒	⊗		
L	_burn	Internal		🔒	⊗		
L	_approve	Private		🔒	⊗		
L	_checkContractOnERC721Received	Private		🔒	⊗		
L	_beforeTokenTransfers	Internal		🔒	⊗		
L	_afterTokenTransfers	Internal		🔒	⊗		
MerkleProof		Library					
L	verify	Internal		🔒			
L	processProof	Internal		🔒			
L	_efficientHash	Private		🔒			
Ownable		Implementation		Context			
L	<Constructor>	Public	!	⊗	NO	!	
L	owner	Public	!		NO	!	

```

| L | renounceOwnership | Public ! |  | onlyOwner |
| L | transferOwnership | Public ! |  | onlyOwner |
| L | _transferOwnership | Internal  |  | |
| | | | |
| **ERC721AQueryable** | Implementation | ERC721A | | |
| L | tokensOfOwner | External ! | | NO! |
| | | | |
| **ISpecialNft** | Interface | | | |
| L | mint | External ! |  | NO! |
| | | | |
| **INftCharacter** | Interface | | | |
| L | mint | External ! |  | NO! |
| | | | |
| **UPWorld** | Implementation | ERC721AQueryable, Ownable | | |
| L | <Constructor> | Public ! |  | ERC721A |
| L | _beforeTokenTransfers | Internal  |  | |
| L | setMaxSupplyforTier1 | External ! |  | onlyOwner |
| L | setMaxSupplyforTier2 | External ! |  | onlyOwner |
| L | setMaxSupplyforTier3 | External ! |  | onlyOwner |
| L | setMaxSupplyforTier4 | External ! |  | onlyOwner |
| L | setMaxSupplyforTier5 | External ! |  | onlyOwner |
| L | setFreeze | External ! |  | onlyOwner |
| L | setNftCharacterAddress | External ! |  | onlyOwner |
| L | setSpecialNftAddress | External ! |  | onlyOwner |
| L | setTierMarker | Internal  |  | |
| L | Reserve | External ! |  | onlyOwner |
| L | burn | Internal  |  | |
| L | burnAndReveal | External ! |  | NO! |
| L | tokenURI | Public ! | | NO! |
| L | setPaused | External ! |  | onlyOwner |
| L | setCost1 | External ! |  | onlyOwner |
| L | setCost2 | External ! |  | onlyOwner |
| L | setCost3 | External ! |  | onlyOwner |
| L | setCost4 | External ! |  | onlyOwner |
| L | setCost5 | External ! |  | onlyOwner |
| L | setMaxTxPerAddress | External ! |  | onlyOwner |
| L | MintTier1 | External ! |  | NO! |
| L | MintTier2 | External ! |  | NO! |
| L | MintTier3 | External ! |  | NO! |
| L | MintTier4 | External ! |  | NO! |
| L | MintTier5 | External ! |  | NO! |
| L | setUriPrefix | External ! |  | onlyOwner |
| L | withdraw | External ! |  | onlyOwner |
| L | _baseURI | Internal  | | |

```

Legend

Symbol	Meaning
:-----:	-----
	Function can modify state
	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. 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 “ Well Secured”.

- ✓ No volatile code.
- ✓ No many high severity issues were found.
- ✓ Low (or very low) level issues have been fixed.

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.