SMART CONTRACT AUDIT REPORT

For

Crazy Whales (CW)

Prepared By: SFI Team

Prepared for: CW team

Prepared on: 29/11/2021

Table of Content

• 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) 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.

- # Overview of the audit

The project has 1 file. It contains approx 2366 lines of Solidity code. Most of the functions and state variables are well commented on using the Nat spec documentation, but that does not create any vulnerability.

- # Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices automatically.

1. Unit tests passing.

2. Compilator warnings;

3. Race Conditions. Reentrancy. Cross-function Race Conditions. Pitfalls in Race Condition solutions;

4. Possible delays in data delivery;

5. Transaction-Ordering Dependence (front running);

6. Timestamp Dependence;

7. Integer Overflow and Underflow;

8. DoS with (unexpected) Revert;

9. DoS with Block Gas Limit;

10. Call Depth Attack. Not relevant in modern ethereum network

11. Methods execution permissions;

12. Oracles calls;

13. Economy model. It’s important to forecast scenarios when a user is provided with additional economic motivation or faced with limitations. If application logic is based on incorrect economy model, the application will not function correctly and participants will incur financial losses. This type of issue is most often found in bonus rewards systems.
14. The impact of the exchange rate on the logic;
15. Private user data leaks.

- # Good things in smart contract

Compiler version is static: -
=> In this file, you have put “pragma solidity 0.8.0;” which is a good way to define the compiler version.

```
pragma solidity 0.8.0;
```

- **Openzeppelin SafeMath library: -**

CW is using openzeppelin SafeMath library it is a good thing. It protects the contract from overflow and underflow.

```
library SafeMath {

    function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            uint256 c = a + b;
            if (c < a) return (false, 0);
            return (true, c);
        }
    }
    function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (b > a) return (false, 0);
            return (true, a - b);
        }
    }
```

- **Openzeppelin Ownable library : -**
  - Here you CW token using openzeppelin ownable library,
    Initializes the contract setting the deployer as the initial owner

```
                abstract contract Ownable is Context {
    address private _owner;

    event OwnershipTransferred(address indexed previousOwner, address indexed
newOwner);

    /**
     * @dev Initializes the contract setting the deployer as the initial owner.
     */
    constructor() {
        _setOwner(_msgSender());
    }

    function owner() public view virtual returns (address) {
        return _owner;
    }
```

- Here you CW token using interface openzeppelin IERC20 which Returns the amount of tokens in existence, symbol, name , owner and etc. based on IERC20 interface

```
interface IERC20 {
    function totalSupply() external view returns (uint256);
    function balanceOf(address account) external view returns (uint256);
    function transfer(address recipient, uint256 amount) external returns (bool);

    function allowance(address owner, address spender) external view returns
(uint256);
```

- Here you CW token using openzeppelin ERC721contract which Inherit Context, ERC165, IERC721, IERC721Metadata (used for Non-Fungible Token Standard), including the Metadata extension, but not including the I Enumerable extension, which is available separately as {ERC721 Enumerable}.

```
contract ERC721 is Context, ERC165, IERC721, IERC721Metadata {
    using Address for address;
    using Strings for uint256;
    string private _name;
    string private _symbol;

    // Mapping from token ID to owner address
    mapping(uint256 => address) private _owners;

    // Mapping owner address to token count
    mapping(address => uint256) private _balances;

    // Mapping from token ID to approved address
    mapping(uint256 => address) private _tokenApprovals;

    // Mapping from owner to operator approvals
    mapping(address => mapping(address => bool)) private _operatorApprovals;
```

- Here you CW using openzeppelin address library which used for Collection of functions related to the address type.

```
library Address {

    function isContract(address account) internal view returns (bool) {
        uint256 size;
        assembly {
            size := extcodesize(account)}
        return size > 0;}
    function sendValue(address payable recipient, uint256 amount) internal {
        require(address(this).balance >= amount, "Address: insufficient balance");

        (bool success, ) = recipient.call{value: amount}("");
        require(success, "Address: unable to send value, recipient may have
reverted");}
```

- ## Critical vulnerabilities found in the contract

There not Critical severity vulnerabilities found

- ## High vulnerabilities found in the contract

There not High severity vulnerabilities found

- ## Medium vulnerabilities found in the contract

There not Medium severity vulnerabilities found

- ## Low vulnerabilities found in the contract

There not Low severity vulnerabilities found

- ## V. Low vulnerabilities found in the contract

#Similar variable names:

```
_name = name_;
_symbol = symbol_;
```

In detail
ERC720 .(string,string) : Variables have very similar names "_symbol" and "symbol_". Note: Modifiers are currently not considered by this static analysis.

Constant/View/Pure functions:

```
function toHexString(uint256 value) internal pure returns (string memory) {
        if (value == 0) {
            return "0x00";
        }
        uint256 temp = value;
        uint256 length = 0;
        while (temp != 0) {
            length++;
            temp >>= 8;
        }
        return toHexString(value, length);
    }
```

In detail
Strings.toHexString(uint256) : Is constant but potentially should not be. Note:
Modifiers are currently not considered by this static analysis.

o Notes

#ERC20:

```
function decimals()
    external
    view
    returns (
      uint8 decimalPlaces
    );
```

In detail
ERC20 contract's "decimals" function should have "uint8" as return type

#Gas Costs:

```
function name() public view virtual override returns (string memory) {
        return _name;
    }
```

In detail
Gas requirement of function CrazyWhales.name is infinite:
If the gas requirement of a function is higher than the block gas limit, it cannot be executed Please avoid loops in your functions or actions that modify large areas of storage
(This includes clearing or copying arrays in storage)

Testing proves:

1- Check for security

fe9c311f64e09856500d8694155182e0a0292a6a82dba9eb84e5713ecfe028fd

File: CrazyW... | Language: solidity | Size: 76266 bytes | Date: 2021-11-29T04:50:12.979Z

Critical	High	Medium	Low	Note
0	0	0	0	2

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

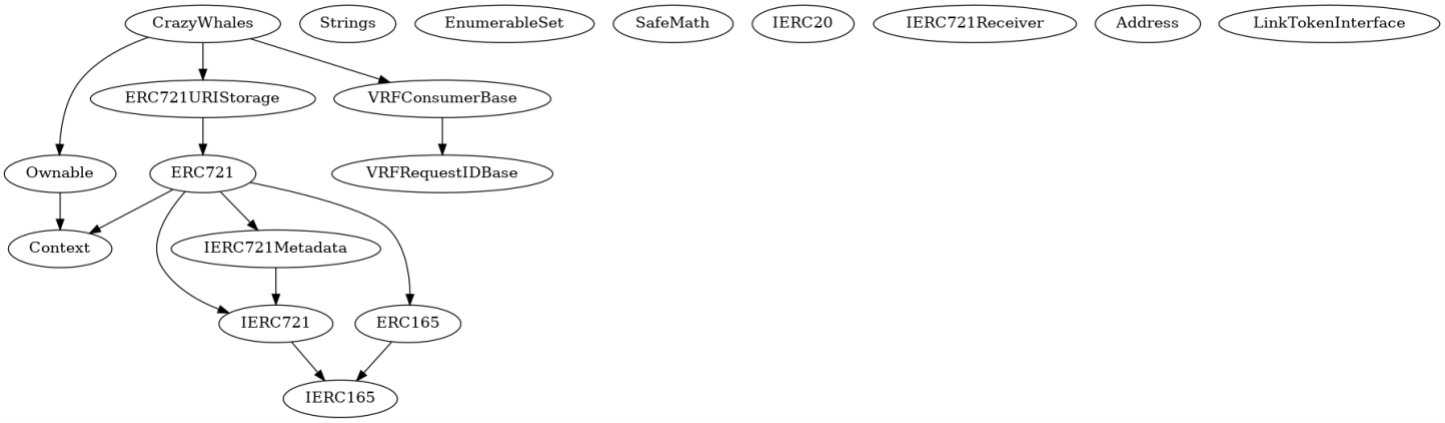
- **ERC20:** 'decimals' should be 'uint8'

Miscellaneous

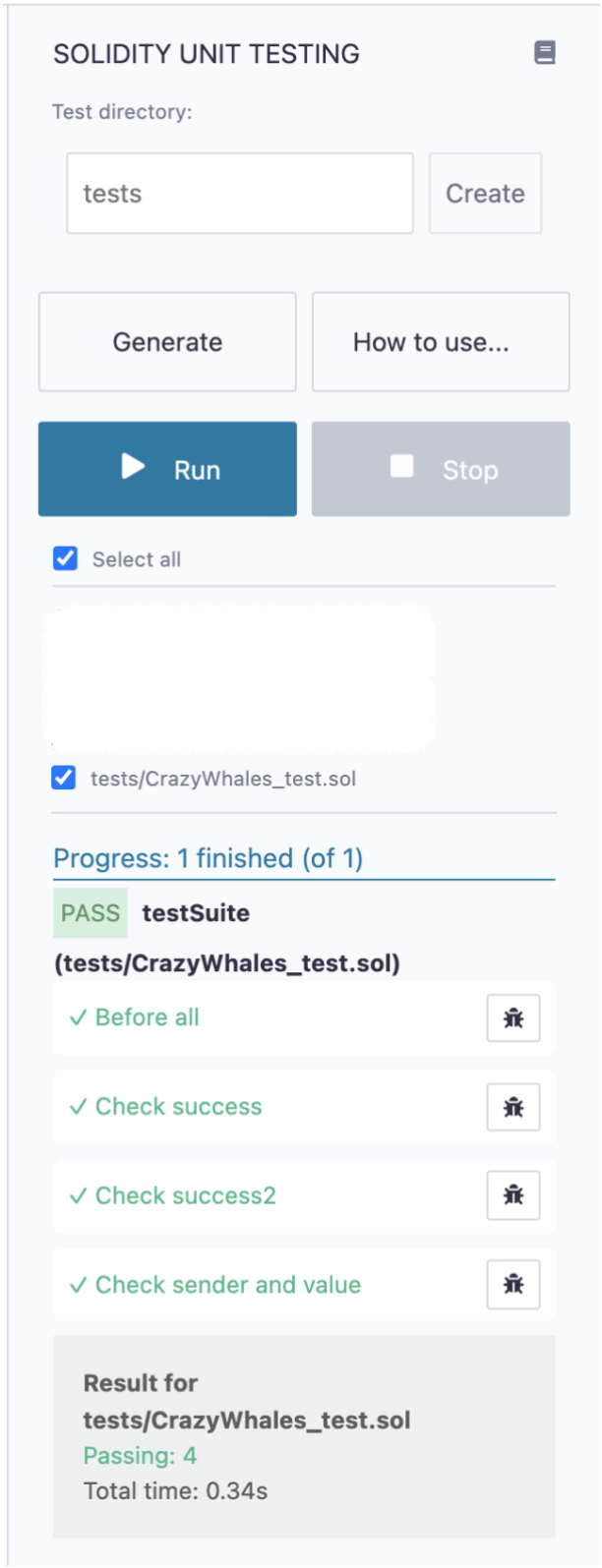
Select Miscellaneous

- **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

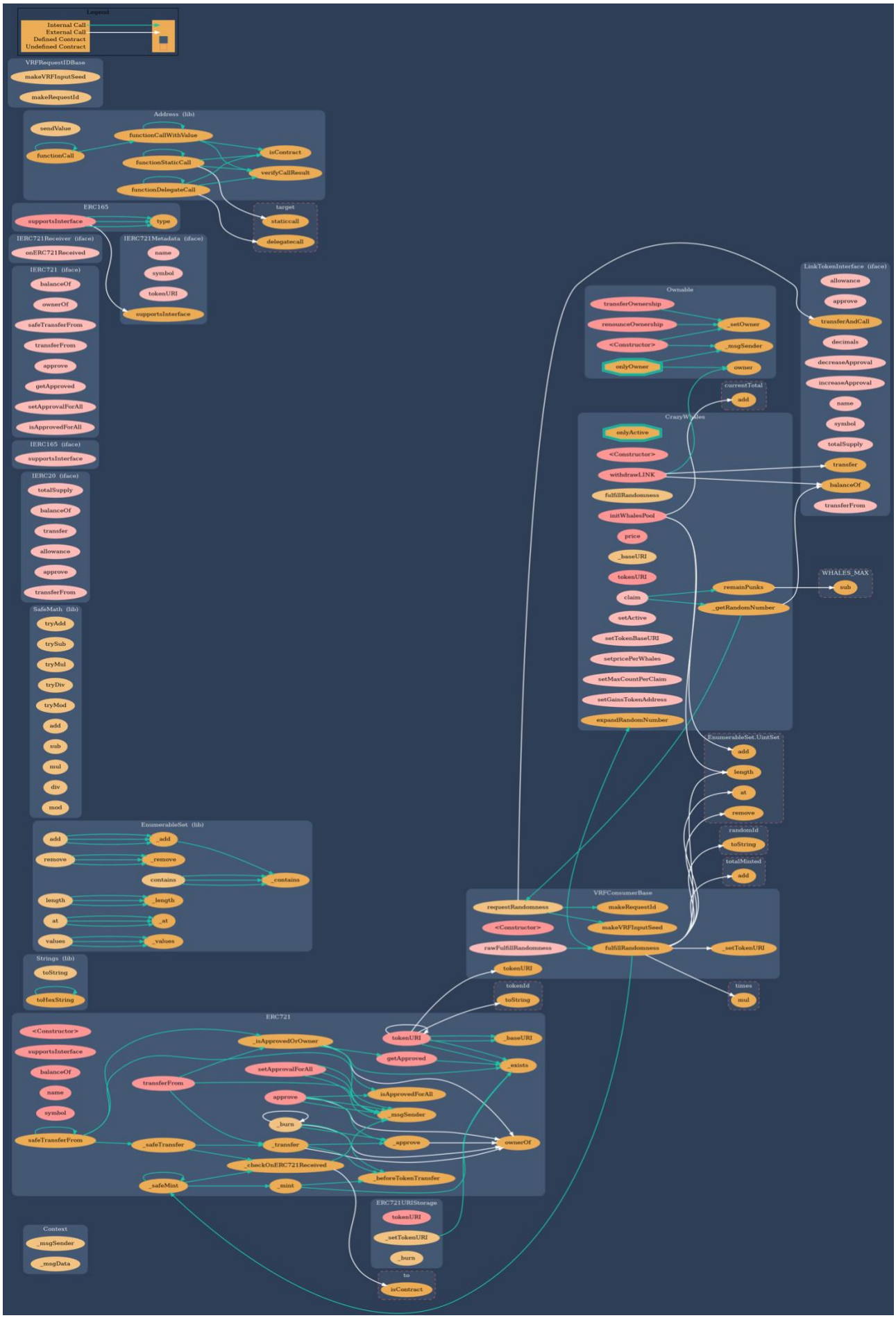
3- Inheritance graph



4- SOLIDITY UNIT TESTING



5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
VRFRequestIDBase
makeVRFInputSeed
makeRequestId
Address (lib)
sendValue
functionCallWithValue
functionCall
functionStaticCall
functionDelegateCall
isContract
verifyCallResult
target
staticcall
delegatecall
ERC165
supportsInterface
type
IERC721Receiver (iface)
onERC721Received
IERC721Metadata (iface)
name
symbol
tokenURI
supportsInterface
IERC721 (iface)
balanceOf
ownerOf
safeTransferFrom
transferFrom
approve
getApproved
setApprovalForAll
isApprovedForAll
IERC165 (iface)
supportsInterface
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
SafeMath (lib)
tryAdd
trySub
tryMul
tryDiv
tryMod
add
sub
mul
div
mod
EnumerableSet (lib)
add
_add
remove
_remove
contains
_contains
length
_length
at
_at
values
_values
Strings (lib)
toString
toHexString
Ownable
transferOwnership
renounceOwnership
<Constructor>
onlyOwner
_setOwner
_msgSender
owner
currentTotal
add
CrazyWhales
onlyActive
<Constructor>
withdrawLINK
fulfillRandomness
initWhalesPool
price
_baseURI
tokenURI
claim
setActive
setTokenBaseURI
setpricePerWhales
setMaxCountPerClaim
setGainsTokenAddress
expandRandomNumber
remainPunks
_getRandomNumber
LinkTokenInterface (iface)
allowance
approve
transferAndCall
decimals
decreaseApproval
increaseApproval
name
symbol
totalSupply
transfer
balanceOf
transferFrom
WHALES_MAX
sub
EnumerableSet.UintSet
add
length
at
remove
randomId
toString
totalMinted
add
VRFConsumerBase
requestRandomness
<Constructor>
rawFulfillRandomness
makeRequestId
makeVRFInputSeed
fulfillRandomness
_setTokenURI
tokenURI
tokenId
toString
times
mul
ERC721
<Constructor>
supportsInterface
balanceOf
name
symbol
safeTransferFrom
_safeTransfer
transferFrom
_isApprovedOrOwner
setApprovalForAll
approve
_burn
transfer
_checkOnERC721Received
_safeMint
_mint
tokenURI
getApproved
isApprovedForAll
_msgSender
_approve
_beforeTokenTransfer
baseURI
_exists
ownerOf
ERC721URIStorage
tokenURI
_setTokenURI
_burn
to
isContract
Context
_msgSender
_msgData

- **Automatic general report**

```
Files Description Table

|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/CrazyWhales.sol |
ff47a1bc62948b559b427337c148543645cc0337 |

 Contracts Description Table

|  Contract  |         Type        |       Bases      |                  |
|
|:----------:|:-------------------:|:----------------:|:----------------:|:--------
-------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |
**Modifiers**  |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | Internal 🔒 |   | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑  |NO❗ |
| └ | owner | Public ❗ |   |NO❗ |
| └ | renounceOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | _setOwner | Private 🔐 | 🛑  | |
||||||
| **Strings** | Library |  |||
| └ | toString | Internal 🔒 |   | |
| └ | toHexString | Internal 🔒 |   | |
| └ | toHexString | Internal 🔒 |   | |
||||||
| **EnumerableSet** | Library |  |||
| └ | _add | Private 🔐 | 🛑  | |
| └ | _remove | Private 🔐 | 🛑  | |
| └ | _contains | Private 🔐 |   | |
| └ | _length | Private 🔐 |   | |
| └ | _at | Private 🔐 |   | |
| └ | _values | Private 🔐 |   | |
| └ | add | Internal 🔒 | 🛑  | |
| └ | remove | Internal 🔒 | 🛑  | |
| └ | contains | Internal 🔒 |   | |
| └ | length | Internal 🔒 |   | |
| └ | at | Internal 🔒 |   | |
| └ | values | Internal 🔒 |   | |
| └ | add | Internal 🔒 | 🛑  | |
| └ | remove | Internal 🔒 | 🛑  | |
| └ | contains | Internal 🔒 |   | |
| └ | length | Internal 🔒 |   | |
| └ | at | Internal 🔒 |   | |
| └ | values | Internal 🔒 |   | |
| └ | add | Internal 🔒 | 🛑  | |
| └ | remove | Internal 🔒 | 🛑  | |
| └ | contains | Internal 🔒 |   | |
```

└	length	Internal 🔒		
└	at	Internal 🔒		
└	values	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 🔒		
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
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❗
IERC721Receiver	Interface			
└	onERC721Received	External ❗	🛑	NO❗
IERC721Metadata	Interface	IERC721		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	tokenURI	External ❗		NO❗
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		

└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResult	Internal 🔒		
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 🔒	🛑	
ERC721URIStorage	Implementation	ERC721		
└	tokenURI	Public ❗		NO❗
└	_setTokenURI	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
LinkTokenInterface	Interface				
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	balanceOf	External ❗		NO❗
└	decimals	External ❗		NO❗
└	decreaseApproval	External ❗	🛑	NO❗
└	increaseApproval	External ❗	🛑	NO❗
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	transferAndCall	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
VRFRequestIDBase	Implementation				
└	makeVRFInputSeed	Internal 🔒		
└	makeRequestId	Internal 🔒		

VRFConsumerBase	Implementation	VRFRequestIDBase		
└	fulfillRandomness	Internal 🔒	🛑	
└	requestRandomness	Internal 🔒	🛑	
└	<Constructor>	Public ❗	🛑	NO❗
└	rawFulfillRandomness	External ❗	🛑	NO❗
CrazyWhales	Implementation	Ownable, ERC721URIStorage, VRFConsumerBase		
└	<Constructor>	Public ❗	🛑	ERC721 VRFConsumerBase
└	claim	External ❗	🛑	onlyActive
└	fulfillRandomness	Internal 🔒	🛑	
└	_getRandomNumber	Private 🔐	🛑	
└	expandRandomNumber	Private 🔐		
└	remainPunks	Public ❗		NO❗
└	price	Public ❗		NO❗
└	_baseURI	Internal 🔒		
└	tokenURI	Public ❗		NO❗
└	initWhalesPool	Public ❗	🛑	onlyOwner
└	setActive	External ❗	🛑	onlyOwner
└	setTokenBaseURI	External ❗	🛑	onlyOwner
└	setpricePerWhales	External ❗	🛑	onlyOwner
└	setMaxCountPerClaim	External ❗	🛑	onlyOwner
└	setGainsTokenAddress	External ❗	🛑	onlyOwner
└	withdrawLINK	Public ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💶	Function is payable

- # Summary of the Audit

According to automatically test, the customer`s solidity smart contract is **Secured**.

The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

The test found 0 critical, 0 high, 0 medium, 0 low, 2 Very low issues, and 2 notes.