Cyberscope

Audit Report

DKeeperNFT

December 2022

Github	https://github.com/Deeplink-Network/Staking
Commit	ab56a7e7cde209bdad1c70a24ce8ce257c04413d
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	DKeeperNFT
Testing Deploy	https://testnet.bscscan.com/token/0xe5518668a2d7e4653ed4bee16460a48fd0be3474

Audit Updates

Initial Audit	12 Dec 2022

Source Files

Filename	SHA256
@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol	da66c17044345dc892d85bd7ddc9745d25df0b3dacfba8f84eb87c60d6e40fe3
@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol	cd823c76cbf5f5b6ef1bda565d58be66c843c37707cd93eb8fb5425deebd6756
@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol	c05b019a0b3bee8f3fac2da7c929f7d665b97d6d046aa35126615fff11205119
@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol	b6adbe9bc075b15cfb4b90f1ae020da4c78e3feada056a4c75b875350285c915
@openzeppelin/contracts-upgradeable/token/ERC721/extensions/IERC721MetadataUpgradeable.sol	78d97961d78c8245d51fef2306c33bbc97edb18cb61159979a75187379e93c86
@openzeppelin/contracts-upgradeable/token/ERC721/IERC721ReceiverUpgradeable.sol	ad8f8e470e39cde97310b6366c8d440aaa000789a7b8c257587fc2075de7cff6
@openzeppelin/contracts-upgradeable/token/ERC721/IERC721Upgradeable.sol	f37f59a19d6d357441a4da0e69a48f793e95dae61f91b2c6ca1a9cf1e224a786
@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol	35fb271561f3dc72e91b3a42c6e40c2bb2e788cd8ca58014ac43f6198b8d32ca
@openzeppelin/contracts-upgradeable/utils/ContextUpgradeable.sol	5fb301961e45cb482fe4e05646d2f529aa449fe0e90c6671475d6a32356fa2d4
@openzeppelin/contracts-upgradeable/utils/introspection/ERC165Upgradeable.sol	fd84e5284eccc479268f0ef36b830019d4f7999ceb7959430d8d8d9e602dd4ef
@openzeppelin/contracts-upgradeable/utils/introspection/IERC165Upgradeable.sol	a39bc026ad6214e9ecd526bd4a1ddf9862d80bd4a9d0d031d9bafa4c3c147c0b
@openzeppelin/contracts-upgradeable/utils/StringsUpgradeable.sol	e7b950eee23563e23989a3b51a1456614a1838084eef1fad04eb2be0bc280f48
contracts/NFT/DKeeper.sol	d288c3e36b31ec77a3633a45b92de866dad60e4da1abfe8ba27507672f2c88e9
contracts/NFT/ERC721AUpgradeable.sol	fffd22e03a5aebdd0e3a04f8bd0acdf381039d524c17302000deee640fb30020

Introduction

The DKeeperNFT contract implements an ERC721AUpgradeable standard. It is an upgradeable NFT contract. The contract applies 0.75% royalty fees for the creators.

There are two ways to mint NFTs, the contract owner mint and public mint.

Contract owner mint

- The contract owner has the ability to mint up to 50 NFTs. The limit cannot be changed.
- The price per NFTs is defined as 2 ETH. The limit cannot be changed.
- The owner does not transfer the corresponding ETH amount to the wallet.

Public Mint

- Any user can mint NFTs publicly.
- The users cannot mint more than a max supply. The max supply can be configured by the contract owner.
- The price per NFT can be between a minimum and maximum limit. The initial value of the minimum limit is 0.5 ETH, it can be configured by the contract owner. The maximum limit is 2 ETH, it cannot be changed.
- Each user cannot mint NFTs for more than a maximum amount worth of ETH. The limit can be configured by the contract owner. The initial value is 1 ETH.
- The price per NFT is defined proportionally to the amount of NFTs that will be minted. For instance, if a user mints 2 NFTs with 1 ETH, then each NFTs price will be defined as 0.5 ETH. In the same line, if a user mints 1 NFT with 1 ETH, then the NFT price will be defined as 1 ETH.

Funds

The funds that are received from the public mint are accumulating to the DKeeper contract. The contract owner has the ability to withdraw the funds by calling the withdraw() method. The funds will be transferred to the treasure wallet. The treasury wallet address can be configured by the contract owner.

Contract Roles

The contract has 2 roles.

EOA Role

The EOA (only wallet) role has the authority to publicMint.

Owner Role

The contract owner has the authority to

- ownerMint
- withdraw
- setBaseURI
- setTreasure
- setPublicMintPriceForEach
- setMaxPublicMintForEach
- setMaxSupply
- togglePause

Users

The contract users have the authority to

- View supportsInterfaces.
- View _startTokenId.
- View _baseURI.
- View royaltyInfo.

Contract Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	MC	Missing Check	unresolved
●	RDS	Redundant Data Structure	unresolved
●	RFI	Redundant Function Issue	unresolved
●	SMT	Stops Mint Transactions	unresolved
●	MEE	Missing Events Emission	unresolved
●	L04	Conformance to Solidity Naming Conventions	unresolved
●	L05	Unused State Variable	unresolved

MC - Missing Check

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L70
Status	unresolved

Description

The contract is processing initializer arguments that have not been properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues.

```
function initialize(
    string memory name_,
    string memory symbol_,
    uint256 maxSupply_,
    address treasure_
) public initializer {
    require(treasure_ != address(0), "Invalid treasure address");
    __ReentrancyGuard_init();
    __ERC721A_init(name_, symbol_);
    __Ownable_init();

    maxPublicMintForEach = 1;

    MAX_SUPPLY = maxSupply_;
    MAX_OWNER_SUPPLY = 50;
    treasure = payable(treasure_);
    priceForEach = 0.5 ether;
    maxPriceForEach = 2 ether;
}
```

Recommendation

The contract should properly check the variables according to the required specifications.

- The variable maxSupply_ should be greater than zero.

RDS - Redundant Data Structure

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L16
Status	unresolved

Description

The contract utilizes a redundant data structure. The contract keeps a RoyaltyInfo registry to keep track of the royalty percentage and the royalty receiver. Since the variable ROYALTY_PERCENT and the _feeDenominator are constant. The royaltyFraction is constant too.

```
uint96 public constant ROYALTY_PERCENT = 75;
function _feeDenominator() internal pure virtual returns (uint96) {
    return 10000;
}
`
```

Hence, the data structure is redundant and increases the code size of the contract unnecessarily.

```
struct RoyaltyInfo {
    address receiver;
    uint96 royaltyFraction;
}
```

Recommendation

Remove the redundant data structure and the functionality around it in order to decrease the code size.

RFI - Redundant Function Issue

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L201
Status	unresolved

Description

The contract is utilizing a function to return a constant value. This code segment could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

```
function _feeDenominator() internal pure virtual returns
(uint96) {
    return 10000;
}
```

Recommendation

The authors are advised to utilize constant variables instead of functions so the runtime will be more performant.

SMT - Stops Mint Transactions

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L216
Status	unresolved

Description

The contract owner has the authority to pause or unpause the transactions for all users including the owner.

```
function togglePause() external onlyOwner {
    if (paused()) {
        _unpause();
    } else {
        _pause();
    }
}

function _beforeTokenTransfers(
    address from,
    address to,
    uint256 startTokenId,
    uint256 quantity
) internal virtual override whenNotPaused {
    super._beforeTokenTransfers(from, to, startTokenId,
quantity);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

MEE - Missing Events Emission

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L123,128,133,154,158
Status	unresolved

Description

Detected missing events for critical access control parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
baseTokenURI
treasure
priceForEach
maxPublicMintForE
MAX_SUPPLY
```

Recommendation

Emit an event for critical parameter changes.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L91,79,185,24,133,185,26,154,146
Status	unresolved

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_amount
_amount
_tokenId
MAX_OWNER_SUPPLY
_treasure
_salePrice
MAX_SUPPLY
_amount
_price
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-conventions.

L05 - Unused State Variable

Criticality	minor / informative
Location	contracts/NFT/DKeeper.sol#L10
Status	unresolved

Description

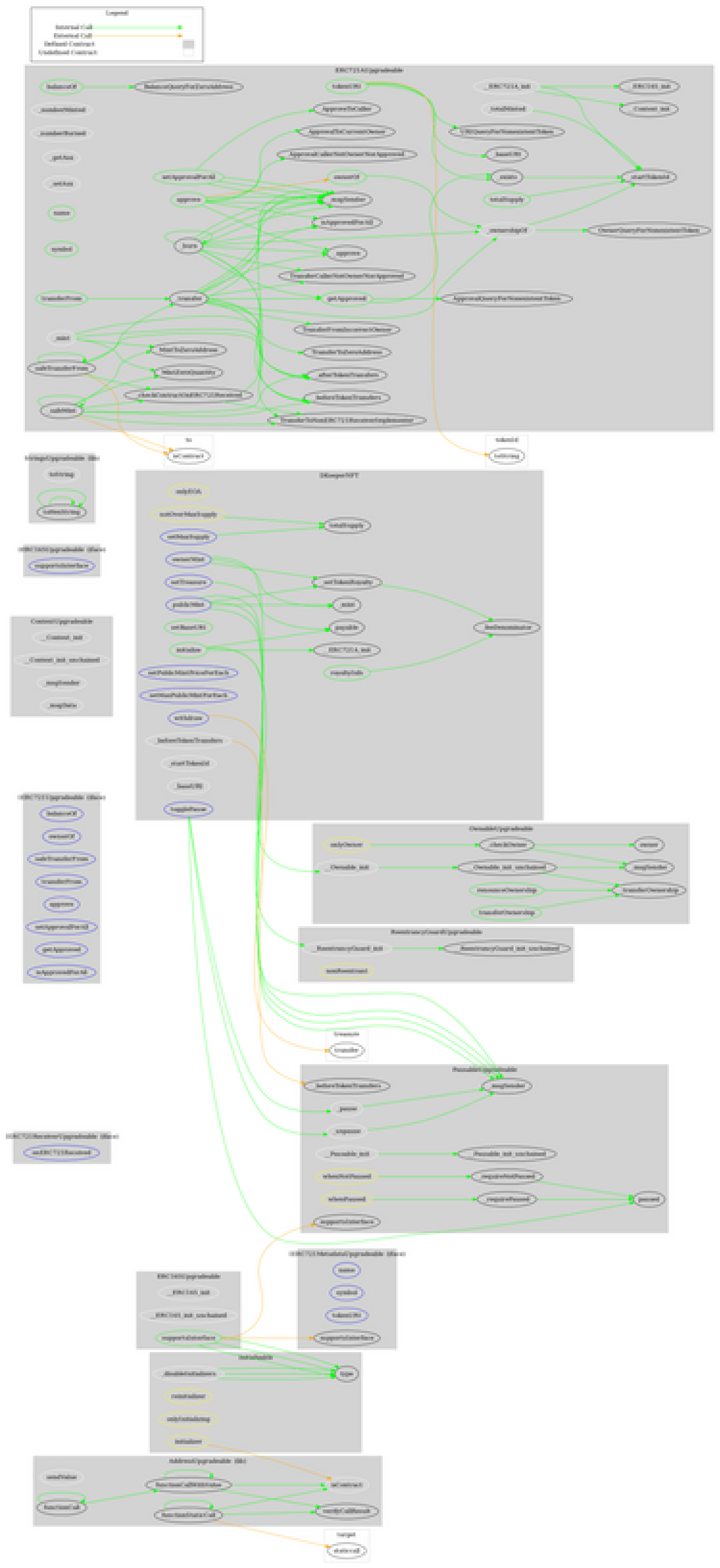
There are segments that contain unused state variables.

```
DKeeperNFT
```

Recommendation

Remove unused state variables.

Contract Flow



Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
OwnableUpgradeable	Implementation	Initializable, ContextUpgradeable		
	__Ownable_init	Internal	✓	onlyInitializing
	__Ownable_init_unchained	Internal	✓	onlyInitializing
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
Initializable	Implementation			
	_disableInitializers	Internal	✓	
PausableUpgradeable	Implementation	Initializable, ContextUpgradeable		
	__Pausable_init	Internal	✓	onlyInitializing
	__Pausable_init_unchained	Internal	✓	onlyInitializing
	paused	Public		-
	_requireNotPaused	Internal		
	_requirePaused	Internal		
	_pause	Internal	✓	whenNotPaused
	_unpause	Internal	✓	whenPaused

ReentrancyGuardUpgradeable	Implementation	Initializable		
	__ReentrancyGuard_init	Internal	✓	onlyInitializing
	__ReentrancyGuard_init_unchained	Internal	✓	onlyInitializing
IERC721MetadataUpgradeable	Interface	IERC721Upgradeable		
	name	External		-
	symbol	External		-
	tokenURI	External		-
IERC721ReceiverUpgradeable	Interface			
	onERC721Received	External	✓	-
IERC721Upgradeable	Interface	IERC165Upgradeable		
	balanceOf	External		-
	ownerOf	External		-
	safeTransferFrom	External	✓	-
	safeTransferFrom	External	✓	-
	transferFrom	External	✓	-
	approve	External	✓	-
	setApprovalForAll	External	✓	-
	getApproved	External		-
	isApprovedForAll	External		-
AddressUpgradeable	Library			
	isContract	Internal		
	sendValue	Internal	✓	

	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	verifyCallResult	Internal		
ContextUpgradeable	Implementation	Initializable		
	__Context_init	Internal	✓	onlyInitializing
	__Context_init_unchained	Internal	✓	onlyInitializing
	_msgSender	Internal		
	_msgData	Internal		
ERC165Upgradeable	Implementation	Initializable, IERC165Upgradeable		
	__ERC165_init	Internal	✓	onlyInitializing
	__ERC165_init_unchained	Internal	✓	onlyInitializing
	supportsInterface	Public		-
IERC165Upgradeable	Interface			
	supportsInterface	External		-
StringsUpgradeable	Library			
	toString	Internal		
	toHexString	Internal		
	toHexString	Internal		
	toHexString	Internal		

DKeeperNFT	Implementation	ERC721AUpgradeable, ReentrancyGuardUpgradeable, OwnableUpgradeable, PausableUpgradeable		
	initialize	Public	✓	initializer
	ownerMint	External	✓	onlyOwner notOverMaxSupply
	publicMint	External	Payable	onlyEOA nonReentrant notOverMaxSupply
	withdraw	External	✓	onlyOwner
	setBaseURI	Public	✓	onlyOwner
	setTreasure	External	✓	onlyOwner
	supportsInterface	Public		-
	setPublicMintPriceForEach	External	✓	onlyOwner
	setMaxPublicMintForEach	External	✓	onlyOwner
	setMaxSupply	External	✓	onlyOwner
	togglePause	External	✓	onlyOwner
	_startTokenId	Internal		
	_baseURI	Internal		
	royaltyInfo	Public		-
	_feeDenominator	Internal		
	_setTokenRoyalty	Internal	✓	
	_beforeTokenTransfers	Internal	✓	whenNotPaused

ERC721AUpgradeable	Implementation	ContextUpgradeable, ERC165Upgradeable, IERC721Upgradeable, IERC721MetadataUpgradeable		
	__ERC721A_init	Internal	✓	onlyInitializing
	_startTokenId	Internal		
	totalSupply	Public		-
	_totalMinted	Internal		
	supportsInterface	Public		-
	balanceOf	Public		-
	_numberMinted	Internal		
	_numberBurned	Internal		
	_getAux	Internal		
	_setAux	Internal	✓	
	_ownershipOf	Internal		
	ownerOf	Public		-
	name	Public		-
	symbol	Public		-
	tokenURI	Public		-
	_baseURI	Internal		
	approve	Public	✓	-
	getApproved	Public		-
	setApprovalForAll	Public	✓	-
	isApprovedForAll	Public		-
	transferFrom	Public	✓	-
	safeTransferFrom	Public	✓	-
	safeTransferFrom	Public	✓	-
	_exists	Internal		

	_safeMint	Internal	✓	
	_safeMint	Internal	✓	
	_mint	Internal	✓	
	_transfer	Private	✓	
	_burn	Internal	✓	
	_burn	Internal	✓	
	_approve	Private	✓	
	_checkContractOnERC721Received	Private	✓	
	_beforeTokenTransfers	Internal	✓	
	_afterTokenTransfers	Internal	✓	

Summary

DKeeperNFT contract implements an nft contract. This audit investigates security issues, business logic concerns, and potential improvements.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope’s prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope’s position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io