Seascape - Mini Miners

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **August 8th, 2022 - August 11th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	08/11/2022	Manuel García
0.2	Document Updates	08/11/2022	Roberto Reigada
0.3	Draft Review	08/12/2022	Gabi Urrutia
1.0	Remediation Plan	08/17/2022	Roberto Reigada
1.1	Remediation Plan Review	08/17/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Manuel García	Halborn	Manuel.Diaz@halborn.com
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Seascape engaged Halborn to conduct a security audit on their smart contracts beginning on August 11th, 2022 and ending on August 12th, 2022. The security assessment was scoped to the smart contract provided in the GitHub repository blocklords/miner-smartcontract

1.2 AUDIT SUMMARY

The team at Halborn was provided a week for the engagement and assigned two full-time security engineers to audit the security of the smart contract. The security engineers are blockchain and smart-contract security experts with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

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

In summary, Halborn identified some security risks that were addressed by Seascape team.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the code and can quickly identify items that do not follow the security best practices. The following phases and associated tools were used during the audit:

- Research into architecture and purpose
- Smart contract manual code review and walkthrough
- Graphing out functionality and contract logic/connectivity/functions (solgraph)
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes
- Manual testing by custom scripts
- Scanning of solidity files for vulnerabilities, security hotspots or bugs. (MythX)
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Testnet deployment (Brownie, Remix IDE)

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.
1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.

3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

IN-SCOPE:

The security assessment was scoped to the following smart contract:

- MineNFTFactory.sol
- MinerGame.sol
- MinerNFT.sol
- NFTTypes.sol
- CrownsToken.sol
- MscpToken.sol

Commit ID:

- 7bdc1d62f738d7f7f26051a0dfb357876ddad087

Fixed commit ID:

- b6fa55a42c3f9c95f2ede398b420f90242dcf914

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	1	1	3

LIKELIHOOD

IMPACT

	(HAL-02)			(HAL-01)
(HAL-03) (HAL-04) (HAL-05)				

EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - SIGNATURE NONCES ARE IMPLEMENTED INCORRECTLY	Medium	SOLVED - 08/17/2022
HAL02 - REENTRANCY IN MINERGAME.EXPORTNFT FUNCTION	Low	SOLVED - 08/17/2022
HAL03 - UNUSED STORAGE POINTER IN MINERGAME.GOLDCHANGETOKEN FUNCTION	Informational	SOLVED - 08/17/2022
HAL04 - POSSIBLE MISUSE OF PUBLIC FUNCTIONS	Informational	SOLVED - 08/17/2022
HAL05 - STATE VARIABLES MISSING CONSTANT MODIFIER	Informational	SOLVED - 08/17/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) SIGNATURE NONCES ARE IMPLEMENTED INCORRECTLY - MEDIUM

Description:

In the MinerGame contract, a nonce state variable is used to prevent signature replay attacks:

Listing 1: MinerGame.sol

```
22 uint256 public nonce;
```

Listing 2: MinerGame.sol (Lines 63,72)

```
55 function importNft(uint256 _nftId, uint8 _v, bytes32 _r, bytes32
↳ _s) external {
56   require(_nftId > 0, "MinerGame: nft Id invalid");
57
58   MineNFT nft = MineNFT(mineNft);
59   require(nft.ownerOf(_nftId) == msg.sender, "MinerGame: Not
↳ mineNft owner");
60
61   {
62     bytes memory prefix     = "\x19Ethereum Signed Message:\n32";
63     bytes32 message         = keccak256(abi.encodePacked(_nftId,
↳ msg.sender, address(this), nonce));
64     bytes32 hash            = keccak256(abi.encodePacked(prefix,
↳ message));
65     address recover         = ecrecover(hash, _v, _r, _s);
66
67     require(recover == verifier, "Verification failed about
↳ stakeToken");
68   }
69
70   nft.safeTransferFrom(msg.sender, address(this), _nftId);
71
72   nonce++;
73
74   PlayerParams storage _player = player[msg.sender];
75   _player.nftId = _nftId;
76   _player.stakeTime = block.timestamp;
```

```
77
78    mineOwners[_nftId] = msg.sender;
79
80    emit ImportNft(msg.sender, _nftId, block.timestamp);
81  }
```

Listing 3: MinerGame.sol (Line 126)

```
114    function goldChangeToken(uint256 _gold, uint8 _v, bytes32 _r,
 ↳ bytes32 _s) external {
115        require(_gold > 0, "MinerGame: The exchange amount must
 ↳ greater than zero");
116
117        uint256 chainId;
118        assembly {
119            chainId := chainid()
120        }
121
122        PlayerParams storage _player = player[msg.sender];
123
124        {
125          bytes memory prefix     = "\x19Ethereum Signed Message:\n32"
 ↳ ;
126          bytes32 message         = keccak256(abi.encodePacked(_gold,
 ↳ msg.sender, nonce, address(this), chainId));
127          bytes32 hash            = keccak256(abi.encodePacked(prefix,
 ↳  message));
128          address recover         = ecrecover(hash, _v, _r, _s);
129
130          require(recover == verifier, "Verification failed about
 ↳ stakeToken");
131        }
132
133        nonce++;
134
135        uint256 _tokenAmount = _gold * MULTIPLIER / ratio;
136        _safeTransfer(token[0], msg.sender, _tokenAmount);
137
138        emit GoldChangeToken(msg.sender, _gold, _tokenAmount, block.
 ↳ timestamp);
139
140    }
141  }
```

This nonce variable is increased each time, the functions importNft() or goldChangeToken() are called. Although, the signer does not really know the order in which the users will call these functions. Hence, if the backend for example generates a signature for a user and this user does not call the function right after that, his signature will be invalid after someone else calls any of those functions.

Risk Level:

Likelihood - 5
Impact - 2

Recommendation:

It is recommended to use a mapping instead of a global counter as a nonce to solve this issue:
mapping(address => uint256)public _nonces;

Remediation Plan:

SOLVED: The Seascape team solved this issue and now uses the suggested mapping as a nonce.

3.2 (HAL-02) REENTRANCY IN MINERGAME.EXPORTNFT FUNCTION - LOW

Description:

In the MinerGame contract, the exportNft() function is used to "unstake" the Mine NFT:

Listing 4: MinerGame.sol (Line 88)

```
84 function exportNft(uint256 _nftId) external {
85   require(mineOwners[_nftId] == msg.sender, "MinerGame: Not the
↳ owner");
86
87   MineNFT nft = MineNFT(mineNft);
88   nft.safeTransferFrom(address(this), msg.sender, _nftId);
89
90   PlayerParams storage _player = player[msg.sender];
91   delete _player.nftId;
92   delete _player.stakeTime;
93
94   delete mineOwners[_nftId];
95
96   emit ExportNft(msg.sender, _nftId, block.timestamp);
97 }
```

As we can see above, the Mine NFT is sent back to the user with a safeTransferFrom() call. This safeTransferFrom() calls check if the receiver is a smart contract and if so, it calls the _checkOnERC721Received hook.

This passed the control flow to the receiver and opens up a reentrancy vulnerability as the user, in this case the smart contract, got the NFT but the state variables player[msg.sender] and mineOwners[_nftId] are still not deleted/updated.

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

It is recommended to add a nonReentrant modifier to the exportNft() function. Other option is updating the exportNft() function as shown below:

Listing 5: MinerGame.sol (Line 96)

```
84 function exportNft(uint256 _nftId) external {
85   require(mineOwners[_nftId] == msg.sender, "MinerGame: Not the
↳ owner");
86
87   MineNFT nft = MineNFT(mineNft);
88
89   PlayerParams storage _player = player[msg.sender];
90   delete _player.nftId;
91   delete _player.stakeTime;
92
93   delete mineOwners[_nftId];
94
95   emit ExportNft(msg.sender, _nftId, block.timestamp);
96   nft.safeTransferFrom(address(this), msg.sender, _nftId);
97 }
```

Remediation Plan:

SOLVED: The Seascape team solved this issue and updated the exportNft() function as suggested.

3.3 (HAL-03) UNUSED STORAGE POINTER IN MINERGAME.GOLDCHANGETOKEN FUNCTION - INFORMATIONAL

Description:

In the MinerGame contract the goldChangeToken() creates a storage pointer to the player mapping, but then it does not make any use of it:

Listing 6: MinerGame.sol (Line 122)

```
114 function goldChangeToken(uint256 _gold, uint8 _v, bytes32 _r,
 ↳ bytes32 _s) external {
115   require(_gold > 0, "MinerGame: The exchange amount must greater
 ↳ than zero");
116
117   uint256 chainId;
118   assembly {
119       chainId := chainid()
120   }
121
122   PlayerParams storage _player = player[msg.sender];
123
124   {
125     bytes memory prefix     = "\x19Ethereum Signed Message:\n32";
126     bytes32 message         = keccak256(abi.encodePacked(_gold,
 ↳ msg.sender, nonce, address(this), chainId));
127     bytes32 hash            = keccak256(abi.encodePacked(prefix,
 ↳ message));
128     address recover         = ecrecover(hash, _v, _r, _s);
129
130     require(recover == verifier, "Verification failed about
 ↳ stakeToken");
131   }
132
133   nonce++;
134
135   uint256 _tokenAmount = _gold * MULTIPLIER / ratio;
136   _safeTransfer(token[0], msg.sender, _tokenAmount);
137
```

```
138    emit GoldChangeToken(msg.sender, _gold, _tokenAmount, block.
 ↳ timestamp);
139
140 }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to remove the pointer to the player mapping to reduce the gas costs of the goldChangeToken() function.

Remediation Plan:

SOLVED: The Seascape team solved this issue and removed the pointer to the player mapping.

3.4 (HAL-04) POSSIBLE MISUSE OF PUBLIC FUNCTIONS - INFORMATIONAL

Description:

In multiple contracts there are functions marked as public but they are never directly called within the same contract or in any of their descendants:

MineNFTFactory.sol
- mint() (MineNFTFactory.sol#34-37)
- setNft() (MineNFTFactory.sol#42-44)
- addAdmin() (MineNFTFactory.sol#47-50)
- renounceAdmin() (MineNFTFactory.sol#53-56)
- addGenerator() (MineNFTFactory.sol#85-88)
- removeGenerator() (MineNFTFactory.sol#91-94)

MineNFT.sol
- mint() (MineNFT.sol#35-46)
- setOwner() (MineNFT.sol#48-50)
- setFactory() (MineNFT.sol#52-54)
- setBaseUri() (MineNFT.sol#56-58)

MinerGame.sol
- withdraw() (MinerGame.sol#188-195)
- addToken() (MinerGame.sol#198-206)
- setScale() (MinerGame.sol#209-212)

CrownsToken.sol
- burn() (CrownsToken.sol#96-98)
- burnFrom() (CrownsToken.sol#111-118)
- name() (CrownsToken.sol#127-129)
- symbol() (CrownsToken.sol#135-137)
- decimals() (CrownsToken.sol#148-150)
- totalSupply() (CrownsToken.sol#155-157)
- balanceOf() (CrownsToken.sol#162-164)
- transfer() (CrownsToken.sol#173-176)

- approve() (CrownsToken.sol#203-206)
- transferFrom() (CrownsToken.sol#217-227)
- increaseAllowance() (CrownsToken.sol#241-244)
- decreaseAllowance() (CrownsToken.sol#260-268)

MscpToken.sol
- burn() (MscpToken.sol#96-98)
- burnFrom() (MscpToken.sol#111-118)
- name() (MscpToken.sol#127-129)
- symbol() (MscpToken.sol#135-137)
- decimals() (MscpToken.sol#148-150)
- totalSupply() (MscpToken.sol#155-157)
- balanceOf() (MscpToken.sol#162-164)
- transfer() (MscpToken.sol#173-176)
- approve() (MscpToken.sol#203-206)
- transferFrom() (MscpToken.sol#217-227)
- increaseAllowance() (MscpToken.sol#241-244)
- decreaseAllowance() (MscpToken.sol#260-268)

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

If the functions are not intended to be called internally or by their descendants, it is better to mark all of these functions as external to reduce the gas costs.

Remediation Plan:

SOLVED: The Seascape team solved this issue and declared the functions as external, reducing the gas costs.

3.5 (HAL-05) STATE VARIABLES MISSING CONSTANT MODIFIER - INFORMATIONAL

Description:

State variables can be declared as constant or immutable. In both cases, the variables cannot be modified after the contract has been constructed. For constant variables, the value has to be fixed at compile-time, while for immutable, it can still be assigned at construction time. The following state variables are missing the constant modifier:

CrownsToken.sol
- Line 30: `uint256 public limitSupply = 11111111110000000000000000; /// 1.1 billion`

MscpToken.sol
- Line 30: `uint256 public limitSupply = 11111111110000000000000000; /// 1.1 billion`

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to add the constant modifier to the state variables mentioned.

Remediation Plan:

SOLVED: The Seascape team solved this issue and declared the suggested state variables as constants.

AUTOMATED TESTING

4.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the smart contracts in scope. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified the smart contracts in the repository and was able to compile them correctly into their abis and binary format, Slither was run against the contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

MineNFTFactory.sol

```
ERC721._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#333-344) ignores return value by _holderTokens[to].add(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#339)
ERC721._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#333-344) ignores return value by _tokenOwners.set(tokenId,to) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#341)
ERC721._burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#356-374) ignores return value by _holderTokens[owner].remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#369)
ERC721._burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#356-374) ignores return value by _tokenOwners.remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#371)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _holderTokens[from].remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#396)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _holderTokens[to].add(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#397)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _tokenOwners.set(tokenId,to) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#399)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

MineNFT.setOwner(address)._owner (contracts/nfts/MineNFT.sol#48) shadows:
        - Ownable._owner (contracts/openzeppelin/contracts/access/Ownable.sol#19) (state variable)
ERC721.constructor(string,string).name (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#93) shadows:
        - ERC721.name() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#122-124) (function)
        - IERC721Metadata.name() (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#16) (function)
ERC721.constructor(string,string).symbol (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#93) shadows:
        - ERC721.symbol() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#129-131) (function)
        - IERC721Metadata.symbol() (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#21) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

MineNFT.setFactory(address) (contracts/nfts/MineNFT.sol#52-54) should emit an event for:
        - factory = _factory (contracts/nfts/MineNFT.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

MineNFT.setFactory(address)._factory (contracts/nfts/MineNFT.sol#52) lacks a zero-check on :
                - factory = _factory (contracts/nfts/MineNFT.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46):
        External calls:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - returnData = to.functionCall(abi.encodeWithSelector(IERC721Receiver(to).onERC721Received.selector,_msgSender(),from,tokenId,_data),ERC721: transfer to non ERC721Receiver implementer) (contracts/openzeppelin/contracts/to
ken/ERC721/ERC721.sol#441-447)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        External calls sending eth:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        State variables written after the call(s):
        - paramsOf[_nftId] = Params(_generation,_quality) (contracts/nfts/MineNFT.sol#41)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46):
        External calls:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - returnData = to.functionCall(abi.encodeWithSelector(IERC721Receiver(to).onERC721Received.selector,_msgSender(),from,tokenId,_data),ERC721: transfer to non ERC721Receiver implementer) (contracts/openzeppelin/contracts/to
ken/ERC721/ERC721.sol#441-447)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        External calls sending eth:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        Event emitted after the call(s):
        - Minted(_to,_nftId,_generation,_quality,block.timestamp) (contracts/nfts/MineNFT.sol#44)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#33)
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#132-135)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['^0.6.7', '^0.6.0', '^0.6.2']
        - 0.6.7 (contracts/factory/MineNFTFactory.sol#2)
        - 0.6.7 (contracts/nfts/MineNFT.sol#2)
        - ^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/access/AccessControl.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/introspection/ERC165.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/introspection/IERC165.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Enumerable.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/Counters.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/Strings.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

AccessControl._setRoleAdmin(bytes32,bytes32) (contracts/openzeppelin/contracts/access/AccessControl.sol#201-204) is never used and should be removed
Address.functionCall(address,bytes) (contracts/openzeppelin/contracts/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#104-106) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#114-117) is never used and should be removed
Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59) is never used and should be removed
Context._msgData() (contracts/openzeppelin/contracts/GSN/Context.sol#20-23) is never used and should be removed
Counters.decrement(Counters.Counter) (contracts/openzeppelin/contracts/utils/Counters.sol#37-39) is never used and should be removed
ERC721._setTokenURI(uint256,string) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#411-414) is never used and should be removed
EnumerableMap._get(EnumerableMap.Map,bytes32) (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#153-155) is never used and should be removed
EnumerableMap.get(EnumerableMap.UintToAddressMap,uint256) (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#227-229) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#219-221) is never used and should be removed
```

```
SafeMath.add(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#29-34) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#103-105) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#119-125) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#139-141) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#151-153) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#77-89) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#46-48) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#60-65) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.6.7 (contracts/factory/MineNFTFactory.sol#2) allows old versions
Pragma version0.6.7 (contracts/nfts/MineNFT.sol#2) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/access/AccessControl.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/introspection/ERC165.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/introspection/IERC165.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Enumerable.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/Counters.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/Strings.sol#3) allows old versions
solc-0.6.7 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (contracts/openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140):
        - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter MineNFTFactory.mint(address,uint256,uint8)._owner (contracts/factory/MineNFTFactory.sol#34) is not in mixedCase
Parameter MineNFTFactory.mint(address,uint256,uint8)._generation (contracts/factory/MineNFTFactory.sol#34) is not in mixedCase
Parameter MineNFTFactory.mint(address,uint256,uint8)._quality (contracts/factory/MineNFTFactory.sol#34) is not in mixedCase
Parameter MineNFTFactory.setNft(address)._nft (contracts/factory/MineNFTFactory.sol#42) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._to (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._generation (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._quality (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.setOwner(address)._owner (contracts/nfts/MineNFT.sol#48) is not in mixedCase
Parameter MineNFT.setFactory(address)._factory (contracts/nfts/MineNFT.sol#52) is not in mixedCase
Parameter MineNFT.setBaseUri(string)._uri (contracts/nfts/MineNFT.sol#56) is not in mixedCase
Parameter ERC721.safeTransferFrom(address,address,uint256,bytes)._data (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#245) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/openzeppelin/contracts/GSN/Context.sol#21)" inContext (contracts/openzeppelin/contracts/GSN/Context.sol#15-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

mint(address,uint256,uint8) should be declared external:
        - MineNFTFactory.mint(address,uint256,uint8) (contracts/factory/MineNFTFactory.sol#34-37)
setNft(address) should be declared external:
        - MineNFTFactory.setNft(address) (contracts/factory/MineNFTFactory.sol#42-44)
addAdmin(address) should be declared external:
        - MineNFTFactory.addAdmin(address) (contracts/factory/MineNFTFactory.sol#47-50)
renounceAdmin() should be declared external:
        - MineNFTFactory.renounceAdmin() (contracts/factory/MineNFTFactory.sol#53-56)
addGenerator(address) should be declared external:
        - MineNFTFactory.addGenerator(address) (contracts/factory/MineNFTFactory.sol#85-88)
removeGenerator(address) should be declared external:
        - MineNFTFactory.removeGenerator(address) (contracts/factory/MineNFTFactory.sol#91-94)
mint(address,uint256,uint8) should be declared external:
        - MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46)
setOwner(address) should be declared external:
        - MineNFT.setOwner(address) (contracts/nfts/MineNFT.sol#48-50)
setFactory(address) should be declared external:
        - MineNFT.setFactory(address) (contracts/nfts/MineNFT.sol#52-54)
setBaseUri(string) should be declared external:
        - MineNFT.setBaseUri(string) (contracts/nfts/MineNFT.sol#56-58)
getRoleMemberCount(bytes32) should be declared external:
        - AccessControl.getRoleMemberCount(bytes32) (contracts/openzeppelin/contracts/access/AccessControl.sol#95-97)
getRoleMember(bytes32,uint256) should be declared external:
        - AccessControl.getRoleMember(bytes32,uint256) (contracts/openzeppelin/contracts/access/AccessControl.sol#111-113)
getRoleAdmin(bytes32) should be declared external:
        - AccessControl.getRoleAdmin(bytes32) (contracts/openzeppelin/contracts/access/AccessControl.sol#121-123)
owner() should be declared external:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/openzeppelin/contracts/access/Ownable.sol#54-57)
supportsInterface(bytes4) should be declared external:
        - ERC165.supportsInterface(bytes4) (contracts/openzeppelin/contracts/introspection/ERC165.sol#35-37)
balanceOf(address) should be declared external:
        - ERC721.balanceOf(address) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#106-110)
name() should be declared external:
        - ERC721.name() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#122-124)
symbol() should be declared external:
        - ERC721.symbol() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#129-131)
tokenURI(uint256) should be declared external:
        - ERC721.tokenURI(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#136-151)
baseURI() should be declared external:
        - ERC721.baseURI() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#158-160)
tokenOfOwnerByIndex(address,uint256) should be declared external:
        - ERC721.tokenOfOwnerByIndex(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#165-167)
totalSupply() should be declared external:
        - ERC721.totalSupply() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#172-175)
tokenByIndex(uint256) should be declared external:
        - ERC721.tokenByIndex(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#180-183)
approve(address,uint256) should be declared external:
        - ERC721.approve(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#188-197)
setApprovalForAll(address,bool) should be declared external:
        - ERC721.setApprovalForAll(address,bool) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#211-216)
transferFrom(address,address,uint256) should be declared external:
        - ERC721.transferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#228-233)
safeTransferFrom(address,address,uint256) should be declared external:
        - ERC721.safeTransferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#238-240)
burn(uint256) should be declared external:
        - ERC721Burnable.burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#20-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

MinerGame.sol

```
MinerGame._safeTransfer(address,address,uint256) (contracts/game/MinerGame.sol#154-172) sends eth to arbitrary user
        Dangerous calls:
        - address(_to).transfer(_amount) (contracts/game/MinerGame.sol#170)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

MinerGame.tokenChangeGold(uint256,uint256) (contracts/game/MinerGame.sol#100-110) ignores return value by _token.transferFrom(msg.sender,address(this),_amount) (contracts/game/MinerGame.sol#107)
MinerGame._safeTransfer(address,address,uint256) (contracts/game/MinerGame.sol#154-172) ignores return value by _rewardToken.transfer(_to,_amount) (contracts/game/MinerGame.sol#162)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer

MinerGame._safeTransfer(address,address,uint256) (contracts/game/MinerGame.sol#154-172) uses a dangerous strict equality:
        - require(bool,string)(_rewardToken.balanceOf(_to) == _beforeBalance + _amount,Invalid transfer) (contracts/game/MinerGame.sol#164)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in MinerGame.exportNft(uint256) (contracts/game/MinerGame.sol#84-97):
        External calls:
        - nft.safeTransferFrom(address(this),msg.sender,_nftId) (contracts/game/MinerGame.sol#88)
        State variables written after the call(s):
        - delete mineOwners[_nftId] (contracts/game/MinerGame.sol#94)
Reentrancy in MinerGame.importNft(uint256,uint8,bytes32,bytes32) (contracts/game/MinerGame.sol#55-81):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_nftId) (contracts/game/MinerGame.sol#70)
        State variables written after the call(s):
        - nonce ++ (contracts/game/MinerGame.sol#72)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

ERC721._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#333-344) ignores return value by _holderTokens[to].add(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#339)
ERC721._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#333-344) ignores return value by _tokenOwners.set(tokenId,to) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#341)
ERC721._burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#356-374) ignores return value by _holderTokens[owner].remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#369)
ERC721._burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#356-374) ignores return value by _tokenOwners.remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#371)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _holderTokens[from].remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#396)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _holderTokens[to].add(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#397)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _tokenOwners.set(tokenId,to) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#399)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

MineNFT.setOwner(address)._owner (contracts/nfts/MineNFT.sol#48) shadows:
        - Ownable._owner (contracts/openzeppelin/contracts/access/Ownable.sol#19) (state variable)
ERC721.constructor(string,string).name (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#93) shadows:
        - ERC721.name() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#122-124) (function)
        - IERC721Metadata.name() (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#16) (function)
ERC721.constructor(string,string).symbol (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#93) shadows:
        - ERC721.symbol() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#129-131) (function)
        - IERC721Metadata.symbol() (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#21) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

MineNFT.setFactory(address) (contracts/nfts/MineNFT.sol#52-54) should emit an event for:
        - factory = _factory (contracts/nfts/MineNFT.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

MineNFT.setFactory(address)._factory (contracts/nfts/MineNFT.sol#52) lacks a zero-check on :
        - factory = _factory (contracts/nfts/MineNFT.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
```

AUTOMATED TESTING

```
Reentrancy in MinerGame.importNft(uint256,uint8,bytes32,bytes32) (contracts/game/MinerGame.sol#55-81):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_nftId) (contracts/game/MinerGame.sol#70)
        State variables written after the call(s):
        - mineOwners[_nftId] = msg.sender (contracts/game/MinerGame.sol#78)
        - _player.nftId = _nftId (contracts/game/MinerGame.sol#75)
        - _player.stakeTime = block.timestamp (contracts/game/MinerGame.sol#76)
Reentrancy in MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46):
        External calls:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - returndata = to.functionCall(abi.encodeWithSelector(IERC721Receiver(to).onERC721Received.selector,_msgSender(),from,tokenId,_data),ERC721: transfer to non ERC721Receiver implementer) (contracts/openzeppelin/contracts/to
ken/ERC721/ERC721.sol#441-447)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        External calls sending eth:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        State variables written after the call(s):
        - paramsOf[_nftId] = Params(_generation,_quality) (contracts/nfts/MineNFT.sol#41)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in MinerGame.exportNft(uint256) (contracts/game/MinerGame.sol#84-97):
        External calls:
        - nft.safeTransferFrom(address(this),msg.sender,_nftId) (contracts/game/MinerGame.sol#88)
        Event emitted after the call(s):
        - ExportNft(msg.sender,_nftId,block.timestamp) (contracts/game/MinerGame.sol#96)
Reentrancy in MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32) (contracts/game/MinerGame.sol#114-140):
        External calls:
        - _safeTransfer(token[0],msg.sender,_tokenAmount) (contracts/game/MinerGame.sol#136)
                - _rewardToken.transfer(_to,_amount) (contracts/game/MinerGame.sol#162)
        External calls sending eth:
        - _safeTransfer(token[0],msg.sender,_tokenAmount) (contracts/game/MinerGame.sol#136)
                - address(_to).transfer(_amount) (contracts/game/MinerGame.sol#170)
        Event emitted after the call(s):
        - GoldChangeToken(msg.sender,_gold,_tokenAmount,block.timestamp) (contracts/game/MinerGame.sol#138)
Reentrancy in MinerGame.importNft(uint256,uint8,bytes32,bytes32) (contracts/game/MinerGame.sol#55-81):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_nftId) (contracts/game/MinerGame.sol#70)
        Event emitted after the call(s):
        - ImportNft(msg.sender,_nftId,block.timestamp) (contracts/game/MinerGame.sol#80)
Reentrancy in MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46):
        External calls:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - returndata = to.functionCall(abi.encodeWithSelector(IERC721Receiver(to).onERC721Received.selector,_msgSender(),from,tokenId,_data),ERC721: transfer to non ERC721Receiver implementer) (contracts/openzeppelin/contracts/to
ken/ERC721/ERC721.sol#441-447)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        External calls sending eth:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        Event emitted after the call(s):
        - Minted(_to,_nftId,_generation,_quality,block.timestamp) (contracts/nfts/MineNFT.sol#44)
Reentrancy in MinerGame.tokenChangeGold(uint256,uint256) (contracts/game/MinerGame.sol#100-110):
        External calls:
        - _token.transferFrom(msg.sender,address(this),_amount) (contracts/game/MinerGame.sol#107)
        Event emitted after the call(s):
        - TokenChangeGold(msg.sender,_amount,block.timestamp) (contracts/game/MinerGame.sol#109)
Reentrancy in MinerGame.withdraw(address,uint256) (contracts/game/MinerGame.sol#185-195):
        External calls:
        - _safeTransfer(_token,owner(),_amount) (contracts/game/MinerGame.sol#192)
                - _rewardToken.transfer(_to,_amount) (contracts/game/MinerGame.sol#162)
        External calls sending eth:
        - _safeTransfer(_token,owner(),_amount) (contracts/game/MinerGame.sol#192)
                - address(_to).transfer(_amount) (contracts/game/MinerGame.sol#170)
        Event emitted after the call(s):
        - Withdraw(_token,_amount,msg.sender,block.timestamp) (contracts/game/MinerGame.sol#194)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32) (contracts/game/MinerGame.sol#114-140) uses assembly
        - INLINE ASM (contracts/game/MinerGame.sol#118-120)
Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#33)
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#132-135)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.6.7', '^0.6.0', '^0.6.2']
        - 0.6.7 (contracts/game/MinerGame.sol#2)
        - 0.6.7 (contracts/nfts/MineNFT.sol#2)
        - ^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/introspection/ERC165.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/introspection/IERC165.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/IERC20.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Enumerable.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/Counters.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/Strings.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/openzeppelin/contracts/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#104-106) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#114-117) is never used and should be removed
Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59) is never used and should be removed
Context._msgData() (contracts/openzeppelin/contracts/GSN/Context.sol#20-23) is never used and should be removed
Counters.decrement(Counters.Counter) (contracts/openzeppelin/contracts/utils/Counters.sol#37-39) is never used and should be removed
ERC721._setTokenURI(uint256,string) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#411-414) is never used and should be removed
EnumerableMap._get(EnumerableMap.Map,bytes32) (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#153-155) is never used and should be removed
EnumerableMap.get(EnumerableMap.UintToAddressMap,uint256) (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#227-229) is never used and should be removed
EnumerableSet.add(EnumerableSet.AddressSet,address) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#147-149) is never used and should be removed
EnumerableSet.at(EnumerableSet.AddressSet,uint256) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#185-187) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet,address) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#164-166) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#218-221) is never used and should be removed
EnumerableSet.length(EnumerableSet.AddressSet) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#171-173) is never used and should be removed
EnumerableSet.remove(EnumerableSet.AddressSet,address) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#157-159) is never used and should be removed
SafeERC20._callOptionalReturn(IERC20,bytes) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#64-74) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#37-46) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#53-56) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#48-51) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#22-24) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#26-28) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#29-34) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#103-105) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#119-125) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#139-141) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#155-158) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#77-89) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#46-48) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#60-65) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.6.7 (contracts/game/MinerGame.sol#2) allows old versions
Pragma version0.6.7 (contracts/nfts/MineNFT.sol#2) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/introspection/ERC165.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/introspection/IERC165.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Enumerable.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/Counters.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/Strings.sol#3) allows old versions
solc-0.6.7 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (contracts/openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140):
        - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter MinerGame.importNft(uint256,uint8,bytes32,bytes32)._nftId (contracts/game/MinerGame.sol#55) is not in mixedCase
Parameter MinerGame.importNft(uint256,uint8,bytes32,bytes32)._v (contracts/game/MinerGame.sol#55) is not in mixedCase
Parameter MinerGame.importNft(uint256,uint8,bytes32,bytes32)._r (contracts/game/MinerGame.sol#55) is not in mixedCase
Parameter MinerGame.importNft(uint256,uint8,bytes32,bytes32)._s (contracts/game/MinerGame.sol#55) is not in mixedCase
Parameter MinerGame.exportNft(uint256)._nftId (contracts/game/MinerGame.sol#84) is not in mixedCase
Parameter MinerGame.tokenChangeGold(uint256,uint256)._typeId (contracts/game/MinerGame.sol#100) is not in mixedCase
Parameter MinerGame.tokenChangeGold(uint256,uint256)._amount (contracts/game/MinerGame.sol#100) is not in mixedCase
Parameter MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32)._gold (contracts/game/MinerGame.sol#114) is not in mixedCase
Parameter MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32)._v (contracts/game/MinerGame.sol#114) is not in mixedCase
Parameter MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32)._r (contracts/game/MinerGame.sol#114) is not in mixedCase
Parameter MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32)._s (contracts/game/MinerGame.sol#114) is not in mixedCase
Parameter MinerGame.checkToken(uint256)._typeId (contracts/game/MinerGame.sol#143) is not in mixedCase
Parameter MinerGame.withdraw(address,uint256)._token (contracts/game/MinerGame.sol#185) is not in mixedCase
Parameter MinerGame.withdraw(address,uint256)._amount (contracts/game/MinerGame.sol#185) is not in mixedCase
Parameter MinerGame.addToken(address)._token (contracts/game/MinerGame.sol#198) is not in mixedCase
Parameter MinerGame.setScale(uint256)._ratio (contracts/game/MinerGame.sol#209) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._to (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._generation (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._quality (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.setOwner(address)._owner (contracts/nfts/MineNFT.sol#48) is not in mixedCase
```

AUTOMATED TESTING

```
Parameter MineNFT.setFactory(address)._factory (contracts/nfts/MineNFT.sol#52) is not in mixedCase
Parameter MineNFT.setBaseUri(string)._uri (contracts/nfts/MineNFT.sol#56) is not in mixedCase
Parameter ERC721.safeTransferFrom(address,address,uint256,bytes)._data (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#245) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/openzeppelin/contracts/GSN/Context.sol#21)" inContext (contracts/openzeppelin/contracts/GSN/Context.sol#15-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Reentrancy in MinerGame.goldChangeToken(uint256,uint8,bytes32,bytes32) (contracts/game/MinerGame.sol#114-140):
        External calls:
        - _safeTransfer(token[0],msg.sender,_tokenAmount) (contracts/game/MinerGame.sol#136)
                - address(_to).transfer(_amount) (contracts/game/MinerGame.sol#170)
        Event emitted after the call(s):
        - GoldChangeToken(msg.sender,_gold,_tokenAmount,block.timestamp) (contracts/game/MinerGame.sol#138)
Reentrancy in MinerGame.withdraw(address,uint256) (contracts/game/MinerGame.sol#155-195):
        External calls:
        - _safeTransfer(_token,owner(),_amount) (contracts/game/MinerGame.sol#192)
                - address(_to).transfer(_amount) (contracts/game/MinerGame.sol#170)
        Event emitted after the call(s):
        - Withdraw(_token,_amount,msg.sender,block.timestamp) (contracts/game/MinerGame.sol#194)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

withdraw(address,uint256) should be declared external:
        - MinerGame.withdraw(address,uint256) (contracts/game/MinerGame.sol#188-195)
addToken(address) should be declared external:
        - MinerGame.addToken(address) (contracts/game/MinerGame.sol#198-206)
setScale(uint256) should be declared external:
        - MinerGame.setScale(uint256) (contracts/game/MinerGame.sol#209-212)
mint(address,uint256,uint8) should be declared external:
        - MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46)
setOwner(address) should be declared external:
        - MineNFT.setOwner(address) (contracts/nfts/MineNFT.sol#48-50)
setFactory(address) should be declared external:
        - MineNFT.setFactory(address) (contracts/nfts/MineNFT.sol#52-54)
setBaseUri(string) should be declared external:
        - MineNFT.setBaseUri(string) (contracts/nfts/MineNFT.sol#56-58)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/openzeppelin/contracts/access/Ownable.sol#54-57)
supportsInterface(bytes4) should be declared external:
        - ERC165.supportsInterface(bytes4) (contracts/openzeppelin/contracts/introspection/ERC165.sol#35-37)
balanceOf(address) should be declared external:
        - ERC721.balanceOf(address) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#106-110)
name() should be declared external:
        - ERC721.name() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#122-124)
symbol() should be declared external:
        - ERC721.symbol() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#129-131)
tokenURI(uint256) should be declared external:
        - ERC721.tokenURI(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#136-151)
baseURI() should be declared external:
        - ERC721.baseURI() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#158-160)
tokenOfOwnerByIndex(address,uint256) should be declared external:
        - ERC721.tokenOfOwnerByIndex(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#165-167)
totalSupply() should be declared external:
        - ERC721.totalSupply() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#172-175)
tokenByIndex(uint256) should be declared external:
        - ERC721.tokenByIndex(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#180-183)
approve(address,uint256) should be declared external:
        - ERC721.approve(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#188-197)
setApprovalForAll(address,bool) should be declared external:
        - ERC721.setApprovalForAll(address,bool) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#211-216)
transferFrom(address,address,uint256) should be declared external:
        - ERC721.transferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#228-233)
safeTransferFrom(address,address,uint256) should be declared external:
        - ERC721.safeTransferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#238-240)
burn(uint256) should be declared external:
        - ERC721Burnable.burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#20-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

MineNFT.sol

```
ERC721._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#333-344) ignores return value by _holderTokens[to].add(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#339)
ERC721._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#333-344) ignores return value by _tokenOwners.set(tokenId,to) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#341)
ERC721._burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#356-374) ignores return value by _holderTokens[owner].remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#369)
ERC721._burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#356-374) ignores return value by _tokenOwners.remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#371)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _holderTokens[from].remove(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#396)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _holderTokens[to].add(tokenId) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#397)
ERC721._transfer(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#387-402) ignores return value by _tokenOwners.set(tokenId,to) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#399)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

MineNFT.setOwner(address)._owner (contracts/nfts/MineNFT.sol#48) shadows:
        - Ownable._owner (contracts/openzeppelin/contracts/access/Ownable.sol#19) (state variable)
ERC721.constructor(string,string).name (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#93) shadows:
        - ERC721.name() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#122-124) (function)
        - IERC721Metadata.name() (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#16) (function)
ERC721.constructor(string,string).symbol (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#93) shadows:
        - ERC721.symbol() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#129-131) (function)
        - IERC721Metadata.symbol() (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#21) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

MineNFT.setFactory(address) (contracts/nfts/MineNFT.sol#52-54) should emit an event for:
        - factory = _factory (contracts/nfts/MineNFT.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

MineNFT.setFactory(address)._factory (contracts/nfts/MineNFT.sol#52) lacks a zero-check on :
                - factory = _factory (contracts/nfts/MineNFT.sol#53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46):
        External calls:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - returnData = to.functionCall(abi.encodeWithSelector(IERC721Receiver(to).onERC721Received.selector,_msgSender(),from,tokenId,_data),ERC721: transfer to non ERC721Receiver implementer) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#441-447)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        External calls sending eth:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        State variables written after the call(s):
        - paramsOf[_nftId] = Params(_generation,_quality) (contracts/nfts/MineNFT.sol#41)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46):
        External calls:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - returnData = to.functionCall(abi.encodeWithSelector(IERC721Receiver(to).onERC721Received.selector,_msgSender(),from,tokenId,_data),ERC721: transfer to non ERC721Receiver implementer) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#441-447)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        External calls sending eth:
        - _safeMint(_to,_nftId) (contracts/nfts/MineNFT.sol#39)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        Event emitted after the call(s):
        - Minted(_to,_nftId,_generation,_quality,block.timestamp) (contracts/nfts/MineNFT.sol#44)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#33)
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#132-135)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.6.7', '^0.6.0', '^0.6.2']
        - 0.6.7 (contracts/nfts/MineNFT.sol#2)
        - ^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/introspection/ERC165.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/introspection/IERC165.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Enumerable.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/Counters.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/utils/Strings.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCall(address,bytes) (contracts/openzeppelin/contracts/utils/Address.sol#79-81) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#104-106) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#114-117) is never used and should be removed
Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59) is never used and should be removed
Context._msgData() (contracts/openzeppelin/contracts/GSN/Context.sol#20-23) is never used and should be removed
Counters.decrement(Counters.Counter) (contracts/openzeppelin/contracts/utils/Counters.sol#37-39) is never used and should be removed
ERC721._setTokenURI(uint256,string) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#411-414) is never used and should be removed
EnumerableMap._get(EnumerableMap.Map,bytes32) (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#153-155) is never used and should be removed
EnumerableMap.get(EnumerableMap.UintToAddressMap,uint256) (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#227-229) is never used and should be removed
EnumerableSet.add(EnumerableSet.AddressSet,address) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#147-149) is never used and should be removed
EnumerableSet.at(EnumerableSet.AddressSet,uint256) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#185-187) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet,address) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#164-166) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#219-221) is never used and should be removed
EnumerableSet.length(EnumerableSet.AddressSet) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#171-173) is never used and should be removed
EnumerableSet.remove(EnumerableSet.AddressSet,address) (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#157-159) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#29-34) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#103-105) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#119-125) is never used and should be removed
```

```
SafeMath.mod(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#139-141) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#155-158) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#77-89) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#46-48) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#60-65) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version0.6.7 (contracts/nfts/MineNFT.sol#2) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/introspection/ERC165.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/introspection/IERC165.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Enumerable.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/token/ERC721/IERC721Metadata.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/Counters.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableMap.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/EnumerableSet.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/utils/Strings.sol#3) allows old versions
solc-0.6.7 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (contracts/openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140):
        - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter MineNFT.mint(address,uint256,uint8)._to (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._generation (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.mint(address,uint256,uint8)._quality (contracts/nfts/MineNFT.sol#35) is not in mixedCase
Parameter MineNFT.setOwner(address)._owner (contracts/nfts/MineNFT.sol#48) is not in mixedCase
Parameter MineNFT.setFactory(address)._factory (contracts/nfts/MineNFT.sol#52) is not in mixedCase
Parameter MineNFT.setBaseUri(string)._uri (contracts/nfts/MineNFT.sol#56) is not in mixedCase
Parameter ERC721.safeTransferFrom(address,address,uint256,bytes)._data (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#245) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/openzeppelin/contracts/GSN/Context.sol#21)" inContext (contracts/openzeppelin/contracts/GSN/Context.sol#15-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

mint(address,uint256,uint8) should be declared external:
        - MineNFT.mint(address,uint256,uint8) (contracts/nfts/MineNFT.sol#35-46)
setOwner(address) should be declared external:
        - MineNFT.setOwner(address) (contracts/nfts/MineNFT.sol#48-50)
setFactory(address) should be declared external:
        - MineNFT.setFactory(address) (contracts/nfts/MineNFT.sol#52-54)
setBaseUri(string) should be declared external:
        - MineNFT.setBaseUri(string) (contracts/nfts/MineNFT.sol#56-58)
owner() should be declared external:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/openzeppelin/contracts/access/Ownable.sol#54-57)
supportsInterface(bytes4) should be declared external:
        - ERC165.supportsInterface(bytes4) (contracts/openzeppelin/contracts/introspection/ERC165.sol#35-37)
balanceOf(address) should be declared external:
        - ERC721.balanceOf(address) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#106-110)
name() should be declared external:
        - ERC721.name() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#122-124)
symbol() should be declared external:
        - ERC721.symbol() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#129-131)
tokenURI(uint256) should be declared external:
        - ERC721.tokenURI(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#136-151)
baseURI() should be declared external:
        - ERC721.baseURI() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#158-160)
tokenOfOwnerByIndex(address,uint256) should be declared external:
        - ERC721.tokenOfOwnerByIndex(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#165-167)
totalSupply() should be declared external:
        - ERC721.totalSupply() (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#172-175)
tokenByIndex(uint256) should be declared external:
        - ERC721.tokenByIndex(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#180-183)
approve(address,uint256) should be declared external:
        - ERC721.approve(address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#188-197)
setApprovalForAll(address,bool) should be declared external:
        - ERC721.setApprovalForAll(address,bool) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#211-216)
transferFrom(address,address,uint256) should be declared external:
        - ERC721.transferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#228-233)
safeTransferFrom(address,address,uint256) should be declared external:
        - ERC721.safeTransferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721.sol#238-240)
burn(uint256) should be declared external:
        - ERC721Burnable.burn(uint256) (contracts/openzeppelin/contracts/token/ERC721/ERC721Burnable.sol#20-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

NFTTypes.sol

No issues found by Slither.

CrownsToken.sol

```
CrownsToken.allowance(address,address).owner (contracts/token/CrownsToken.sol#185) shadows:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37) (function)
CrownsToken._approve(address,address,uint256).owner (contracts/token/CrownsToken.sol#354) shadows:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#33)
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#132-135)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['0.6.7', '^0.6.0', '^0.6.2']
        - ^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/IERC20.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3)
        - 0.6.7 (contracts/token/CrownsToken.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) is never used and should be removed
Address.functionCall(address,bytes) (contracts/openzeppelin/contracts/utils/Address.sol#79-81) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/openzeppelin/contracts/utils/Address.sol#89-91) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#104-106) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#114-117) is never used and should be removed
Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) is never used and should be removed
Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59) is never used and should be removed
Context._msgData() (contracts/openzeppelin/contracts/GSN/Context.sol#20-23) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#103-105) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#119-125) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#139-141) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#155-158) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#77-89) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version0.6.7 (contracts/token/CrownsToken.sol#2) allows old versions
solc-0.6.7 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (contracts/openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140):
        - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter CrownsToken.addBridge(address)._bridge (contracts/token/CrownsToken.sol#57) is not in mixedCase
Parameter CrownsToken.removeBridge(address)._bridge (contracts/token/CrownsToken.sol#66) is not in mixedCase
Constant CrownsToken._name (contracts/token/CrownsToken.sol#23) is not in UPPER_CASE_WITH_UNDERSCORES
Constant CrownsToken._symbol (contracts/token/CrownsToken.sol#24) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
Redundant expression "this (contracts/openzeppelin/contracts/GSN/Context.sol#21)" inContext (contracts/openzeppelin/contracts/GSN/Context.sol#15-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

CrownsToken.constructor() (contracts/token/CrownsToken.sol#53-55) uses literals with too many digits:
        - _mint(msg.sender,10000000 * SCALER) (contracts/token/CrownsToken.sol#54)
CrownsToken.slitherConstructorVariables() (contracts/token/CrownsToken.sol#14-387) uses literals with too many digits:
        - limitSupply = 11111111110000000000000000 (contracts/token/CrownsToken.sol#30)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

CrownsToken.limitSupply (contracts/token/CrownsToken.sol#30) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

owner() should be declared external:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/openzeppelin/contracts/access/Ownable.sol#54-57)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (contracts/openzeppelin/contracts/access/Ownable.sol#63-67)
burn(uint256) should be declared external:
        - CrownsToken.burn(uint256) (contracts/token/CrownsToken.sol#96-98)
burnFrom(address,uint256) should be declared external:
        - CrownsToken.burnFrom(address,uint256) (contracts/token/CrownsToken.sol#111-118)
name() should be declared external:
        - CrownsToken.name() (contracts/token/CrownsToken.sol#127-129)
symbol() should be declared external:
        - CrownsToken.symbol() (contracts/token/CrownsToken.sol#135-137)
decimals() should be declared external:
        - CrownsToken.decimals() (contracts/token/CrownsToken.sol#148-150)
totalSupply() should be declared external:
        - CrownsToken.totalSupply() (contracts/token/CrownsToken.sol#155-157)
balanceOf(address) should be declared external:
        - CrownsToken.balanceOf(address) (contracts/token/CrownsToken.sol#162-164)
transfer(address,uint256) should be declared external:
        - CrownsToken.transfer(address,uint256) (contracts/token/CrownsToken.sol#173-176)
approve(address,uint256) should be declared external:
        - CrownsToken.approve(address,uint256) (contracts/token/CrownsToken.sol#203-206)
transferFrom(address,address,uint256) should be declared external:
        - CrownsToken.transferFrom(address,address,uint256) (contracts/token/CrownsToken.sol#217-227)
increaseAllowance(address,uint256) should be declared external:
        - CrownsToken.increaseAllowance(address,uint256) (contracts/token/CrownsToken.sol#241-244)
decreaseAllowance(address,uint256) should be declared external:
        - CrownsToken.decreaseAllowance(address,uint256) (contracts/token/CrownsToken.sol#260-268)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

MscpToken.sol

```
MscpToken.allowance(address,address).owner (contracts/token/MscpToken.sol#185) shadows:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37) (function)
MscpToken._approve(address,address,uint256).owner (contracts/token/MscpToken.sol#354) shadows:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#33)
Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) uses assembly
        - INLINE ASM (contracts/openzeppelin/contracts/utils/Address.sol#132-135)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
        - Version used: ['^0.6.7', '^0.6.0', '^0.6.2']
        - ^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/IERC20.sol#3)
        - ^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3)
        - 0.6.7 (contracts/token/MscpToken.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140) is never used and should be removed
Address.functionCall(address,bytes) (contracts/openzeppelin/contracts/utils/Address.sol#79-81) is never used and should be removed
Address.functionCall(address,bytes,string) (contracts/openzeppelin/contracts/utils/Address.sol#89-91) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#104-106) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#114-117) is never used and should be removed
Address.isContract(address) (contracts/openzeppelin/contracts/utils/Address.sol#26-35) is never used and should be removed
Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59) is never used and should be removed
Context._msgData() (contracts/openzeppelin/contracts/GSN/Context.sol#20-23) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#103-105) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#119-125) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#139-141) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/openzeppelin/contracts/math/SafeMath.sol#155-158) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/openzeppelin/contracts/math/SafeMath.sol#77-89) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.6.0 (contracts/openzeppelin/contracts/GSN/Context.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/access/Ownable.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/math/SafeMath.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.6.2 (contracts/openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version0.6.7 (contracts/token/MscpToken.sol#2) allows old versions
solc-0.6.7 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/openzeppelin/contracts/utils/Address.sol#53-59):
        - (success) = recipient.call{value: amount}() (contracts/openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (contracts/openzeppelin/contracts/utils/Address.sol#119-140):
        - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter MscpToken.addBridge(address)._bridge (contracts/token/MscpToken.sol#57) is not in mixedCase
Parameter MscpToken.removeBridge(address)._bridge (contracts/token/MscpToken.sol#66) is not in mixedCase
Constant MscpToken._name (contracts/token/MscpToken.sol#23) is not in UPPER_CASE_WITH_UNDERSCORES
Constant MscpToken._symbol (contracts/token/MscpToken.sol#24) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/openzeppelin/contracts/GSN/Context.sol#21)" inContext (contracts/openzeppelin/contracts/GSN/Context.sol#15-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

MscpToken.constructor() (contracts/token/MscpToken.sol#53-55) uses literals with too many digits:
        - _mint(msg.sender,10000000 * SCALER) (contracts/token/MscpToken.sol#54)
MscpToken.slitherConstructorVariables() (contracts/token/MscpToken.sol#14-387) uses literals with too many digits:
        - limitSupply = 11111111110000000000000000 (contracts/token/MscpToken.sol#30)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

MscpToken.limitSupply (contracts/token/MscpToken.sol#30) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

owner() should be declared external:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37)
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (contracts/openzeppelin/contracts/access/Ownable.sol#54-57)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (contracts/openzeppelin/contracts/access/Ownable.sol#63-67)
burn(uint256) should be declared external:
        - MscpToken.burn(uint256) (contracts/token/MscpToken.sol#96-98)
burnFrom(address,uint256) should be declared external:
        - MscpToken.burnFrom(address,uint256) (contracts/token/MscpToken.sol#111-118)
name() should be declared external:
        - MscpToken.name() (contracts/token/MscpToken.sol#127-129)
symbol() should be declared external:
        - MscpToken.symbol() (contracts/token/MscpToken.sol#135-137)
decimals() should be declared external:
        - MscpToken.decimals() (contracts/token/MscpToken.sol#148-150)
totalSupply() should be declared external:
        - MscpToken.totalSupply() (contracts/token/MscpToken.sol#155-157)
balanceOf(address) should be declared external:
        - MscpToken.balanceOf(address) (contracts/token/MscpToken.sol#162-164)
transfer(address,uint256) should be declared external:
        - MscpToken.transfer(address,uint256) (contracts/token/MscpToken.sol#173-176)
approve(address,uint256) should be declared external:
        - MscpToken.approve(address,uint256) (contracts/token/MscpToken.sol#203-206)
transferFrom(address,address,uint256) should be declared external:
        - MscpToken.transferFrom(address,address,uint256) (contracts/token/MscpToken.sol#217-227)
increaseAllowance(address,uint256) should be declared external:
        - MscpToken.increaseAllowance(address,uint256) (contracts/token/MscpToken.sol#241-244)
decreaseAllowance(address,uint256) should be declared external:
        - MscpToken.decreaseAllowance(address,uint256) (contracts/token/MscpToken.sol#260-268)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
```

- The reentrancies flagged were checked individually and are all false positives except the one mentioned in the findings.
- No major issues found by Slither.

THANK YOU FOR CHOOSING

// HALBORN