Seascape - Moonscape

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **August 22nd, 2022 - August 30th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	08/22/2022	Luis Arroyo
0.2	Draft Review	09/01/2022	Kubilay Onur Gungor
0.3	Draft Review	09/01/2022	Gabi Urrutia
1.0	Remediation Plan	09/07/2022	Luis Arroyo
1.1	Remediation Plan Review	09/07/2022	Kubilay Onur Gungor
1.2	Remediation Plan Review	09/07/2022	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
Kubilay Onur Gungor	Halborn	Kubilay.Gungor@halborn.com

Luis Arroyo	Halborn	Luis.Arroyo@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

SeaScape engaged Halborn to conduct a security audit on their smart contracts beginning on August 22nd, 2022 and ending on August 30th, 2022 . The security assessment was scoped to the smart contracts provided to the Halborn team in the GitHub repository Seastarinteractive/moonscape-smartcontracts/

1.2 AUDIT SUMMARY

The team at Halborn was provided one week for the engagement and assigned a full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert 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 mostly addressed by the 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 bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of 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
- 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

The security assessment was scoped to the following Seastarinteractive/moonscape-smartcontracts/

- /contracts/beta/MoonscapeBeta.sol
- /contracts/defi/MoonscapeDefi.sol
- /contracts/defi/Stake.sol
- /contracts/defi/StakeNftForChain.sol
- /contracts/game/MoonscapeGame.sol
- /contracts/mscp/MscpToken.sol
- /contracts/mscp/MscpVesting.sol
- /contracts/mscp/MscpVesting5M.sol
- /contracts/mscp/MscpVesting30M.sol
- /contracts/nfts/CityNft.sol
- /contracts/nfts/RoverNft.sol
- /contracts/nfts/SeascapeNft.sol

Commit ID: 4a56c7d2b9d209ec63541ca9aa32692941e66371

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	3	0	3

LIKELIHOOD

IMPACT

	(HAL-02)			
		(HAL-03)		
(HAL-04)				(HAL-01)
(HAL-06)	(HAL-05)			

EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - SIGNATURE NONCES ARE IMPLEMENTED INCORRECTLY	Medium	SOLVED - 09/07/2022
HAL02 - MISSING SIGNATURE VERIFICATION	Medium	SOLVED - 09/07/2022
HAL03 - MINTERS CANT BE UNSET	Medium	SOLVED - 09/07/2022
HAL04 - UNUSED PARAMETERS	Informational	ACKNOWLEDGED
HAL05 - FUNCTION STATE CAN BE RESTRICTED	Informational	ACKNOWLEDGED
HAL06 - PRAGMA VERSION	Informational	ACKNOWLEDGED

FINDINGS & TECH DETAILS

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

Description:

In the MoonscapeDefi and StakeNftForChain contracts, a nonce state variable is used to prevent signature replay attacks:

Listing 1: MoonscapeDefi.sol

```
23 uint256 public nonce;
```

Listing 2: MoonscapeDefi.sol (Lines 146,153,163)

```
132     function stakeToken(uint _stakeId, uint _cityId, uint
 ↳ _buildingId, uint _amount, uint8 v, bytes32[2] calldata sig)
 ↳ external {
133         TokenStaking storage tokenStaking = tokenStakings[_stakeId
 ↳ ];
134 
135         // todo
136         // validate the session id
137         bytes32 stakeKey = stakeKeyOf(tokenStaking.sessionId,
 ↳ _stakeId);
138 
139         require(isActive(stakeKey), "session not active");
140 
141         //validate stake id
142         require(_stakeId <= stakeId,"do not have this stakeId");
143 
144         {
145             bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
146             bytes32 message         = keccak256(abi.encodePacked(
 ↳ _stakeId, tokenStaking.sessionId, _cityId, _buildingId, nonce, msg
 ↳ .sender));
147             bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
148             address recover         = ecrecover(hash, v, sig[0],
 ↳ sig[1]);
149 
```

```
150             // require(recover == verifier, "Verification failed
 ↳ about stakeToken");
151         }
152
153         ++nonce;
154
155         deposit(stakeKey, msg.sender, _amount);
156
157         IERC20 token = IERC20(tokenStaking.stakeToken);
158
159         require(token.balanceOf(msg.sender) >= _amount, "Not
 ↳ enough token to stake");
160
161         token.safeTransferFrom(msg.sender, address(this), _amount)
 ↳ ;
162
163         emit StakeToken(msg.sender, tokenStaking.sessionId,
 ↳ _stakeId, _cityId, _buildingId, _amount, nonce);
164     }
```

Listing 3: MoonscapeDefi.sol (Lines 195,202)

```
180     function importNft(uint _stakeId, uint _cityId, uint
 ↳ _buildingId, uint _scapeNftId, uint8 _v, bytes32[2] calldata sig)
 ↳ external {
181         TokenStaking storage tokenStaking = tokenStakings[_stakeId
 ↳ ];
182
183         // validate the session id
184         bytes32 stakeKey = stakeKeyOf(tokenStaking.sessionId,
 ↳ _stakeId);
185
186         require(isActive(stakeKey), "session not active");
187
188         //validate stake id
189         require(_stakeId <= stakeId, "do not have this stakeId");
190
191         require(nft.ownerOf(_scapeNftId) == msg.sender, "not owned
 ↳ ");
192
193         {
194             bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
```

```
195             bytes32 message         = keccak256(abi.encodePacked(
 ↳ tokenStaking.sessionId, _stakeId, _cityId, _buildingId,
 ↳ _scapeNftId, nonce, msg.sender));
196             bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
197             address recover         = ecrecover(hash, _v, sig[0],
 ↳ sig[1]);
198 
199             // require(recover == verifier, "Verification failed
 ↳ about stakeNft");
200         }
201 
202         ++nonce;
203 
204         nft.safeTransferFrom(msg.sender, 0
 ↳ x000000000000000000000000000000000000dEaD, _scapeNftId);
205 
206         emit ImportNft(msg.sender, tokenStaking.sessionId,
 ↳ _stakeId, _cityId, _buildingId, _scapeNftId, block.timestamp);
207 
208     }
```

Listing 4: MoonscapeDefi.sol (Lines 251,258)

```
243     function verifyBonus(uint _sessionId, uint _stakeId, uint
 ↳ _cityId, uint _buildingId, uint _bonusPercent, uint8 _v, bytes32
 ↳ _r, bytes32 _s) internal returns(bool) {
244 
245         bytes32 stakeKey = stakeKeyOf(_sessionId, _stakeId);
246 
247         require(receiveBonus[stakeKey][msg.sender] == false, "
 ↳ already rewarded");
248 
249         {
250             bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
251             bytes32 message         = keccak256(abi.encodePacked(
 ↳ _sessionId, _stakeId, _cityId, _buildingId, _bonusPercent, nonce,
 ↳ msg.sender));
252             bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
253             address recover         = ecrecover(hash, _v, _r, _s);
254 
```

```
255             // require(recover == verifier, "Verification failed
 ↳ about getBonus");
256         }
257
258         ++nonce;
259
260         return true;
261     }
```

Listing 5: StakeNftForChain.sol (Line 65)

```
50      function importNft(uint _stakeId, uint _cityId, uint
 ↳ _buildingId, uint _scapeNftId, uint8 _v, bytes32[2] calldata sig)
 ↳ external {
51
52          require(isActive(sessionId), "session not active");
53
54          require(nft.ownerOf(_scapeNftId) == msg.sender, "not owner
 ↳ ");
55
56          {
57              bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
58              bytes32 message         = keccak256(abi.encodePacked(
 ↳ sessionId, _stakeId, _cityId, _buildingId, _scapeNftId, nonce, msg
 ↳ .sender));
59              bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
60              address recover         = ecrecover(hash, _v, sig[0],
 ↳ sig[1]);
61
62              // require(recover == verifier, "Verification failed
 ↳ about stakeNft");
63          }
64
65          ++nonce;
66
67          nft.safeTransferFrom(msg.sender, 0
 ↳ x000000000000000000000000000000000000dEaD, _scapeNftId);
68
69          emit ImportNft(msg.sender, sessionId, _stakeId, _cityId,
 ↳ _buildingId, _scapeNftId, block.timestamp);
70      }
```

This nonce variable is increased when stakeToken(), importNft() or verifyBonus() functions 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 now uses a nonce mapping for each user. This issue were fixed in the commit ID cdc174452dae98665bfda883dca9c7ee46dda50f

3.2 (HAL-02) MISSING SIGNATURE VERIFICATION - MEDIUM

Description:

In the MoonscapeDefi and StakeNftForChain contracts, some functions use the ecrecover() function when staking or importing NFTs or calculating bonuses. This ecrecover() function is a handy Solidity function that allows the smart contract to validate that an expected party properly signs incoming data.

The mentioned functions use the ecrecover() but there is no validation of the obtained address by this function.

Listing 6: MoonscapeDefi.sol (Line 150)

```
132     function stakeToken(uint _stakeId, uint _cityId, uint
 ↳ _buildingId, uint _amount, uint8 v, bytes32[2] calldata sig)
 ↳ external {
133         TokenStaking storage tokenStaking = tokenStakings[_stakeId
 ↳ ];
134 
135         // todo
136         // validate the session id
137         bytes32 stakeKey = stakeKeyOf(tokenStaking.sessionId,
 ↳ _stakeId);
138 
139         require(isActive(stakeKey), "session not active");
140 
141         //validate stake id
142         require(_stakeId <= stakeId,"do not have this stakeId");
143 
144         {
145             bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
146             bytes32 message         = keccak256(abi.encodePacked(
 ↳ _stakeId, tokenStaking.sessionId, _cityId, _buildingId, nonce, msg
 ↳ .sender));
147             bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
148             address recover         = ecrecover(hash, v, sig[0],
```

```
 ↳ sig[1]);
149
150             // require(recover == verifier, "Verification failed
 ↳ about stakeToken");
151         }
152
153         ++nonce;
154
155         deposit(stakeKey, msg.sender, _amount);
156
157         IERC20 token = IERC20(tokenStaking.stakeToken);
158
159         require(token.balanceOf(msg.sender) >= _amount, "Not
 ↳ enough token to stake");
160
161         token.safeTransferFrom(msg.sender, address(this), _amount)
 ↳ ;
162
163         emit StakeToken(msg.sender, tokenStaking.sessionId,
 ↳ _stakeId, _cityId, _buildingId, _amount, nonce);
164     }
```

Listing 7: MoonscapeDefi.sol (Line 199)

```
180     function importNft(uint _stakeId, uint _cityId, uint
 ↳ _buildingId, uint _scapeNftId, uint8 _v, bytes32[2] calldata sig)
 ↳ external {
181         TokenStaking storage tokenStaking = tokenStakings[_stakeId
 ↳ ];
182
183         // validate the session id
184         bytes32 stakeKey = stakeKeyOf(tokenStaking.sessionId,
 ↳ _stakeId);
185
186         require(isActive(stakeKey), "session not active");
187
188         //validate stake id
189         require(_stakeId <= stakeId, "do not have this stakeId");
190
191         require(nft.ownerOf(_scapeNftId) == msg.sender, "not owned
 ↳ ");
192
193         {
194             bytes memory prefix     = "\x19Ethereum Signed Message
```

```
 ↳ :\n32";
195             bytes32 message         = keccak256(abi.encodePacked(
 ↳ tokenStaking.sessionId, _stakeId, _cityId, _buildingId,
 ↳ _scapeNftId, nonce, msg.sender));
196             bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
197             address recover         = ecrecover(hash, _v, sig[0],
 ↳ sig[1]);
198
199             // require(recover == verifier, "Verification failed
 ↳ about stakeNft");
200         }
201
202         ++nonce;
203
204         nft.safeTransferFrom(msg.sender, 0
 ↳ x000000000000000000000000000000000dEaD, _scapeNftId);
205
206         emit ImportNft(msg.sender, tokenStaking.sessionId,
 ↳ _stakeId, _cityId, _buildingId, _scapeNftId, block.timestamp);
207
208     }
```

Listing 8: MoonscapeDefi.sol (Line 255)

```
243     function verifyBonus(uint _sessionId, uint _stakeId, uint
 ↳ _cityId, uint _buildingId, uint _bonusPercent, uint8 _v, bytes32
 ↳ _r, bytes32 _s) internal returns(bool) {
244
245         bytes32 stakeKey = stakeKeyOf(_sessionId, _stakeId);
246
247         require(receiveBonus[stakeKey][msg.sender] == false, "
 ↳ already rewarded");
248
249         {
250             bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
251             bytes32 message         = keccak256(abi.encodePacked(
 ↳ _sessionId, _stakeId, _cityId, _buildingId, _bonusPercent, nonce,
 ↳ msg.sender));
252             bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
253             address recover         = ecrecover(hash, _v, _r, _s);
254
```

```
255             // require(recover == verifier, "Verification failed
 ↳ about getBonus");
256         }
257
258         ++nonce;
259
260         return true;
261     }
```

Listing 9: StakeNftForChain.sol (Line 65)

```
50      function importNft(uint _stakeId, uint _cityId, uint
 ↳ _buildingId, uint _scapeNftId, uint8 _v, bytes32[2] calldata sig)
 ↳ external {
51
52          require(isActive(sessionId), "session not active");
53
54          require(nft.ownerOf(_scapeNftId) == msg.sender, "not owner
 ↳ ");
55
56          {
57              bytes memory prefix     = "\x19Ethereum Signed Message
 ↳ :\n32";
58              bytes32 message         = keccak256(abi.encodePacked(
 ↳ sessionId, _stakeId, _cityId, _buildingId, _scapeNftId, nonce, msg
 ↳ .sender));
59              bytes32 hash            = keccak256(abi.encodePacked(
 ↳ prefix, message));
60              address recover         = ecrecover(hash, _v, sig[0],
 ↳ sig[1]);
61
62              // require(recover == verifier, "Verification failed
 ↳ about stakeNft");
63          }
64
65          ++nonce;
66
67          nft.safeTransferFrom(msg.sender, 0
 ↳ x000000000000000000000000000000000000dEaD, _scapeNftId);
68
69          emit ImportNft(msg.sender, sessionId, _stakeId, _cityId,
 ↳ _buildingId, _scapeNftId, block.timestamp);
70      }
```

Risk Level:

Likelihood - 2
Impact - 5

Recommendation:

It is recommended to validate the recovered address with the one allowed to execute the called function.

Remediation Plan:

SOLVED: The SeaScape Team corrected the recovered address verification. This issue were fixed in the commit ID cdc174452dae98665bfda883dca9c7ee46dda50f

3.3 (HAL-03) MINTERS CANT BE UNSET - MEDIUM

Description:

The unsetMinter() function from RoverNft.sol smart contract does not correctly implement the verification to check if an account is a minter. This statement prevents minters from being removed.

Code Location:

Listing 10: Rover.Nft.sol (Line 46)

```
45      function unsetMinter(address _minter) public onlyOwner {
46          require(!minters[_minter], "already a minter");
47
48          delete minters[_minter];
49
50          emit UnsetMinter(_minter);
51      }
```

Proof of Concept:

Steps to replicate this issue:
- add a minter, for example user1
- try to remove user1 from minters

Listing 11: RoverTest.js

```
1       it("ROVER :: unset minters", async () => {
2           await rover.setMinter(user1.address);
3           await rover.unsetMinter(user1.address);
4       });
```

Listing 12: Output

```
    ROVER :: unset minters:
     Error: VM Exception while processing transaction: reverted
↳ with reason string 'already a minter'
    at RoverNft.unsetMinter (contracts/nfts/RoverNft.sol:46)
```

Risk Level:

Likelihood - 3
Impact - 3

Recommendation:

It is recommended to correctly validate if the user is a minter before removing it.

Remediation Plan:

SOLVED: The SeaScape Team corrected the verification to determine whether an account is a minter or not. This issue were fixed in the commit ID cdc174452dae98665bfda883dca9c7ee46dda50f

3.4 (HAL-04) UNUSED PARAMETERS - INFORMATIONAL

Description:

There are functions whose parameters are never used in some smart contracts. These parameters do not affect the code.

Code Location:

- operator, from, tokenId, data (MoonscapeDefi.sol#393,394,395,396)
- operator, from, tokenId, data (MoonscapeGame.sol#275,276,277.278)
- operator, from, tokenId, data (StakeNftForChain.sol#98)
- amount (MMscpToken.sol#95)

Risk Level:

Likelihood - 1
Impact - 2

Recommendation:

All parameters given to a function should affect the state of the code. The operator, from, tokenId, and data from onERC721Received() functions should implement any logic with that parameters, but they are not used for any contract state operations.

Remediation Plan:

ACKNOWLEDGED: The SeaScape Team acknowledged this finding.

3.5 (HAL-05) FUNCTION STATE CAN BE RESTRICTED - INFORMATIONAL

Description:

The state mutability of some functions could be restricted to pure in order to save gas.

Code Location:

Listing 13: MoonscapeDefi.sol

```
367     function stakeKeyOf(uint _sessionId, uint _stakeId) public
 ↳ virtual returns(bytes32) {
368         return keccak256(abi.encodePacked(_sessionId, _stakeId));
369     }
```

Listing 14: MscpVesting.sol

```
140     function getAvailableTokens(
141         bool _strategicInvestor,
142         uint256 _timePassed,
143         uint256 _remainingCoins
144     )
145         internal
146         view
147         returns(uint)
148     {
```

Listing 15: MscpVesting.sol

```
161     function getBonus(bool _strategicInvestor) internal view
 ↳ returns(uint) {
162         if(_strategicInvestor)
163             return 2000000 * 10**18; // 2 mil is released on day
 ↳ one
164         return 1500000 * 10**18; // 1.5 mil is released on day one
165     }
```

Listing 16: MscpToken.sol

```
95      function burn(uint256 amount) public {
96          require(false, "Only burnFrom is allowed");
97      }
```

Risk Level:

Likelihood - 2
Impact - 1

Recommendation:

It is recommended to restrict the state of the mentioned functions to pure.

Remediation Plan:

ACKNOWLEDGED: The SeaScape Team acknowledged this finding.

3.6 (HAL-06) PRAGMA VERSION - INFORMATIONAL

Description:

The moonScape smart contracts use the pragma version 0.6.7 released on May 4, 2020. latest pragma version 0.8.16 released on August 8, 2022, solves different issues. It is also noticeable that Solidity versions after 0.8.0 also implement default overflow protection on arithmetic operations.

Reference: Solidity Releases

Code Location:

note: All moonScape smart contracts implement same pragma version.

Listing 17: MoonscapeGame.sol

```
1 pragma solidity 0.6.7;
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

Updating the pragma version used in MoonScape smart contracts is recommended to versions above or equal to 0.8.0.

Remediation Plan:

ACKNOWLEDGED: The SeaScape Team acknowledged this finding.

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:

```
MoonscapeDefi._safeTransfer(address,address,uint256) (contracts/defi/MoonscapeDefi.sol#295-313) sends eth to arbitrary user
        Dangerous calls:
        - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

MoonscapeDefi._safeTransfer(address,address,uint256) (contracts/defi/MoonscapeDefi.sol#295-313) ignores return value by _rewardToken.transfer(_to,_amount) (contracts/
defi/MoonscapeDefi.sol#303)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer

Stake._reward(bytes32,address) (contracts/defi/Stake.sol#210-235) uses a dangerous strict equality:
        - interest == 0 (contracts/defi/Stake.sol#219)
MoonscapeDefi._safeTransfer(address,address,uint256) (contracts/defi/MoonscapeDefi.sol#295-313) uses a dangerous strict equality:
        - require(bool,string)(_rewardToken.balanceOf(_to) == _beforBalance + _amount,Invalid transfer) (contracts/defi/MoonscapeDefi.sol#305)
Stake.claimable(bytes32,address) (contracts/defi/Stake.sol#237-260) uses a dangerous strict equality:
        - sessionCap == period.endTime && staker.rewardClaimedTime >= sessionCap (contracts/defi/Stake.sol#252)
Stake.validStakePeriodParams(bytes32,uint256,uint256,uint256) (contracts/defi/Stake.sol#45-51) uses a dangerous strict equality:
        - require(bool,string)(stakePeriods[key].startTime == 0,STAKE_TOKEN: period_exists) (contracts/defi/Stake.sol#49)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities

Reentrancy in MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/game/MoonscapeGame.sol#181-202):
        External calls:
        - nft.safeTransferFrom(msg.sender,dead,_scapeNftId) (contracts/game/MoonscapeGame.sol#197)
        State variables written after the call(s):
        - buildingScapeBurns[_sessionId][msg.sender][_buildingId] = _scapeNftId (contracts/game/MoonscapeGame.sol#199)
Reentrancy in MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32) (contracts/game/MoonscapeGame.sol#204-224):
        External calls:
        - nft.safeTransferFrom(msg.sender,dead,_scapeNftId) (contracts/game/MoonscapeGame.sol#219)
        State variables written after the call(s):
        - connectionScapeBurns[_sessionId][msg.sender] = _scapeNftId (contracts/game/MoonscapeGame.sol#221)
Reentrancy in MoonscapeGame.exportCity(uint256) (contracts/game/MoonscapeGame.sol#147-156):
        External calls:
        - nft.safeTransferFrom(address(this),msg.sender,_id) (contracts/game/MoonscapeGame.sol#151)
        State variables written after the call(s):
        - delete cityOwners[_id] (contracts/game/MoonscapeGame.sol#153)
Reentrancy in MoonscapeGame.exportRover(uint256) (contracts/game/MoonscapeGame.sol#244-253):
        External calls:
        - nft.safeTransferFrom(address(this),msg.sender,_id) (contracts/game/MoonscapeGame.sol#248)
        State variables written after the call(s):
        - delete roverOwners[_id] (contracts/game/MoonscapeGame.sol#250)
Reentrancy in MoonscapeGame.unstakeForMoondust(uint256) (contracts/game/MoonscapeGame.sol#112-126):
        External calls:
        - require(bool,string)(_token.transfer(msg.sender,_amount),Failed to transfer token from contract to user) (contracts/game/MoonscapeGame.sol#121)
        State variables written after the call(s):
        - _balance.stakeAmount = _balance.stakeAmount.sub(_amount) (contracts/game/MoonscapeGame.sol#123)
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/ope
nzeppelin/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/openz
eppelin/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/openz
eppelin/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(tokenI
d) (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) (c
ontracts/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) (con
tracts/openzeppelin/contracts/token/ERC721/ERC721.sol#399)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

MscpToken.allowance(address,address).owner (contracts/mscp/MscpToken.sol#184) shadows:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37) (function)
MscpToken._approve(address,address,uint256).owner (contracts/mscp/MscpToken.sol#353) shadows:
        - Ownable.owner() (contracts/openzeppelin/contracts/access/Ownable.sol#35-37) (function)
SeascapeNft.setOwner(address)._owner (contracts/nfts/SeascapeNft.sol#61) shadows:
        - Ownable._owner (contracts/openzeppelin/contracts/access/Ownable.sol#19) (state variable)
ERC20.constructor(string,string).name (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#57) shadows:
        - ERC20.name() (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#66-68) (function)
ERC20.constructor(string,string).symbol (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#57) shadows:
        - ERC20.symbol() (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#74-76) (function)
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

SeascapeNft.setFactory(address) (contracts/nfts/SeascapeNft.sol#65-67) should emit an event for:
        - factory = _factory (contracts/nfts/SeascapeNft.sol#66)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

StakeNftForChain.startSession(uint256,uint256,address)._verifier (contracts/defi/StakeNftForChain.sol#37) lacks a zero-check on :
                - verifier = _verifier (contracts/defi/StakeNftForChain.sol#43)
SeascapeNft.setFactory(address)._factory (contracts/nfts/SeascapeNft.sol#65) lacks a zero-check on :
                - factory = _factory (contracts/nfts/SeascapeNft.sol#66)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Modifier Migrations.restricted() (contracts/Migrations.sol#7-9) does not always execute _; or revertReference: https://github.com/crytic/slither/wiki/Detector-Documen
tation#incorrect-modifier
```

AUTOMATED TESTING

```
Reentrancy in MoonscapeDefi.giveBonus(uint256,uint256,address,uint256) (contracts/defi/MoonscapeDefi.sol#265-282):
        External calls:
        - res = rewardToken.transferFrom(owner(),msg.sender,_bonusPercent) (contracts/defi/MoonscapeDefi.sol#271)
        State variables written after the call(s):
        - receiveBonus[stakeKey][msg.sender] = true (contracts/defi/MoonscapeDefi.sol#277)
Reentrancy in MoonscapeGame.importCity(uint256) (contracts/game/MoonscapeGame.sol#135-145):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_id) (contracts/game/MoonscapeGame.sol#141)
        State variables written after the call(s):
        - cityOwners[_id] = msg.sender (contracts/game/MoonscapeGame.sol#142)
Reentrancy in MoonscapeGame.importRover(uint256) (contracts/game/MoonscapeGame.sol#232-242):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_id) (contracts/game/MoonscapeGame.sol#238)
        State variables written after the call(s):
        - roverOwners[_id] = msg.sender (contracts/game/MoonscapeGame.sol#239)
Reentrancy in SeascapeNft.mint(address,uint256,uint8) (contracts/nfts/SeascapeNft.sol#48-59):
        External calls:
        - _safeMint(_to,_tokenId) (contracts/nfts/SeascapeNft.sol#51)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
                - 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)
        External calls sending eth:
        - _safeMint(_to,_tokenId) (contracts/nfts/SeascapeNft.sol#51)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        State variables written after the call(s):
        - paramsOf[_tokenId] = Params(_generation,_quality) (contracts/nfts/SeascapeNft.sol#53)
Reentrancy in MoonscapeGame.purchaseMoondust(uint256) (contracts/game/MoonscapeGame.sol#86-97):
        External calls:
        - require(bool,string)(_token.transferFrom(msg.sender,feeTo,_amount),transfer of tokens to contract failed) (contracts/game/MoonscapeGame.sol#91)
        State variables written after the call(s):
        - _balance.totalSpent = _amount.add(_balance.totalSpent) (contracts/game/MoonscapeGame.sol#94)
Reentrancy in MoonscapeGame.stakeForMoondust(uint256) (contracts/game/MoonscapeGame.sol#99-110):
        External calls:
        - require(bool,string)(_token.transferFrom(msg.sender,address(this),_amount),transfer of tokens to contract failed) (contracts/game/MoonscapeGame.sol#104)
        State variables written after the call(s):
        - _balance.stakeAmount = _amount.add(_balance.stakeAmount) (contracts/game/MoonscapeGame.sol#107)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/game/MoonscapeGame.sol#181-202):
        External calls:
        - nft.safeTransferFrom(msg.sender,dead,_scapeNftId) (contracts/game/MoonscapeGame.sol#197)
        Event emitted after the call(s):
        - BurnScapeForBuilding(msg.sender,_scapeNftId,_sessionId,_cityId,_buildingId) (contracts/game/MoonscapeGame.sol#201)
Reentrancy in MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32) (contracts/game/MoonscapeGame.sol#204-224):
        External calls:
        - nft.safeTransferFrom(msg.sender,dead,_scapeNftId) (contracts/game/MoonscapeGame.sol#219)
        Event emitted after the call(s):
        - BurnScapeForConnection(msg.sender,_scapeNftId,_sessionId) (contracts/game/MoonscapeGame.sol#223)
```

```
Reentrancy in MoonscapeGame.exportCity(uint256) (contracts/game/MoonscapeGame.sol#147-156):
        External calls:
        - nft.safeTransferFrom(address(this),msg.sender,_id) (contracts/game/MoonscapeGame.sol#151)
        Event emitted after the call(s):
        - ExportCity(msg.sender,_id,block.timestamp) (contracts/game/MoonscapeGame.sol#155)
Reentrancy in MoonscapeGame.exportRover(uint256) (contracts/game/MoonscapeGame.sol#244-253):
        External calls:
        - nft.safeTransferFrom(address(this),msg.sender,_id) (contracts/game/MoonscapeGame.sol#248)
        Event emitted after the call(s):
        - ExportRover(msg.sender,_id,block.timestamp) (contracts/game/MoonscapeGame.sol#252)
Reentrancy in MoonscapeDefi.giveBonus(uint256,uint256,address,uint256) (contracts/defi/MoonscapeDefi.sol#265-282):
        External calls:
        - res = rewardToken.transferFrom(owner(),msg.sender,_bonusPercent) (contracts/defi/MoonscapeDefi.sol#271)
        Event emitted after the call(s):
        - GiveBonus(_sessionId,_stakeId,_bonusPercent,_rewardToken,msg.sender,block.timestamp) (contracts/defi/MoonscapeDefi.sol#279)
Reentrancy in MoonscapeGame.importCity(uint256) (contracts/game/MoonscapeGame.sol#135-145):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_id) (contracts/game/MoonscapeGame.sol#141)
        Event emitted after the call(s):
        - ImportCity(msg.sender,_id,block.timestamp) (contracts/game/MoonscapeGame.sol#144)
Reentrancy in MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/defi/MoonscapeDefi.sol#180-208):
        External calls:
        - nft.safeTransferFrom(msg.sender,0x000000000000000000000000000000000000dEaD,_scapeNftId) (contracts/defi/MoonscapeDefi.sol#204)
        Event emitted after the call(s):
        - ImportNft(msg.sender,tokenStaking.sessionId,_stakeId,_cityId,_buildingId,_scapeNftId,block.timestamp) (contracts/defi/MoonscapeDefi.sol#206)
Reentrancy in StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/defi/StakeNftForChain.sol#50-70):
        External calls:
        - nft.safeTransferFrom(msg.sender,0x000000000000000000000000000000000000dEaD,_scapeNftId) (contracts/defi/StakeNftForChain.sol#67)
        Event emitted after the call(s):
        - ImportNft(msg.sender,sessionId,_stakeId,_cityId,_buildingId,_scapeNftId,block.timestamp) (contracts/defi/StakeNftForChain.sol#69)
Reentrancy in MoonscapeGame.importRover(uint256) (contracts/game/MoonscapeGame.sol#232-242):
        External calls:
        - nft.safeTransferFrom(msg.sender,address(this),_id) (contracts/game/MoonscapeGame.sol#238)
        Event emitted after the call(s):
        - ImportRover(msg.sender,_id,block.timestamp) (contracts/game/MoonscapeGame.sol#241)
Reentrancy in MoonscapeBeta.lock(uint256) (contracts/beta/MoonscapeBeta.sol#43-56):
        External calls:
        - session.token.safeTransferFrom(msg.sender,address(this),session.requiredAmount) (contracts/beta/MoonscapeBeta.sol#53)
        Event emitted after the call(s):
        - Transfer(msg.sender,true) (contracts/beta/MoonscapeBeta.sol#55)
```

```
Reentrancy in CityNft.mint(uint256,uint8,address) (contracts/nfts/CityNft.sol#24-35):
        External calls:
        - _safeMint(_to,_tokenId) (contracts/nfts/CityNft.sol#31)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
                - 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)
        External calls sending eth:
        - _safeMint(_to,_tokenId) (contracts/nfts/CityNft.sol#31)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        Event emitted after the call(s):
        - Minted(_to,_tokenId,_category,block.timestamp) (contracts/nfts/CityNft.sol#33)
Reentrancy in RoverNft.mint(uint256,uint8,address) (contracts/nfts/RoverNft.sol#24-35):
        External calls:
        - _safeMint(_to,_tokenId) (contracts/nfts/RoverNft.sol#31)
                - 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,_tokenId) (contracts/nfts/RoverNft.sol#31)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        Event emitted after the call(s):
        - Minted(_to,_tokenId,_type,block.timestamp) (contracts/nfts/RoverNft.sol#33)
Reentrancy in SeascapeNft.mint(address,uint256,uint8) (contracts/nfts/SeascapeNft.sol#48-59):
        External calls:
        - _safeMint(_to,_tokenId) (contracts/nfts/SeascapeNft.sol#51)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
                - 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)
        External calls sending eth:
        - _safeMint(_to,_tokenId) (contracts/nfts/SeascapeNft.sol#51)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
        Event emitted after the call(s):
        - Minted(_to,_tokenId,_generation,_quality) (contracts/nfts/SeascapeNft.sol#57)
Reentrancy in MoonscapeGame.mintCity(uint256,uint8,uint8,bytes32,bytes32) (contracts/game/MoonscapeGame.sol#158-173):
        External calls:
        - require(bool,string)(nft.mint(_id,_category,msg.sender),Failed to mint city) (contracts/game/MoonscapeGame.sol#170)
        Event emitted after the call(s):
        - MintCity(msg.sender,_id,_category) (contracts/game/MoonscapeGame.sol#172)
Reentrancy in MoonscapeGame.mintRover(uint256,uint8,uint8,bytes32,bytes32) (contracts/game/MoonscapeGame.sol#255-270):
        External calls:
        - require(bool,string)(nft.mint(_id,_type,msg.sender),Failed to mint rover) (contracts/game/MoonscapeGame.sol#267)
        Event emitted after the call(s):
        - MintRover(msg.sender,_id,_type) (contracts/game/MoonscapeGame.sol#269)
Reentrancy in MoonscapeGame.purchaseMoondust(uint256) (contracts/game/MoonscapeGame.sol#86-97):
        External calls:
        - require(bool,string)(_token.transferFrom(msg.sender,feeTo,_amount),transfer of tokens to contract failed) (contracts/game/MoonscapeGame.sol#91)
        Event emitted after the call(s):
        - Spent(msg.sender,_amount,block.timestamp,_balance.totalSpent) (contracts/game/MoonscapeGame.sol#96)
Reentrancy in MoonscapeGame.stakeForMoondust(uint256) (contracts/game/MoonscapeGame.sol#99-110):
        External calls:
        - require(bool,string)(_token.transferFrom(msg.sender,address(this),_amount),transfer of tokens to contract failed) (contracts/game/MoonscapeGame.sol#104)
        Event emitted after the call(s):
        - Stake(msg.sender,_amount,block.timestamp,_balance.stakeAmount) (contracts/game/MoonscapeGame.sol#109)
Reentrancy in MoonscapeDefi.stakeToken(uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/defi/MoonscapeDefi.sol#132-164):
        External calls:
        - deposit(stakeKey,msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#155)
                - _rewardToken.transfer(_to,_amount) (contracts/defi/MoonscapeDefi.sol#303)
        - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/defi/MoonscapeDefi.sol#161)
        External calls sending eth:
        - deposit(stakeKey,msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#155)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        Event emitted after the call(s):
        - StakeToken(msg.sender,tokenStaking.sessionId,_stakeId,_cityId,_buildingId,_amount,nonce) (contracts/defi/MoonscapeDefi.sol#163)
Reentrancy in MoonscapeDefi.tokenChangeMoondust(uint256,uint256,uint256) (contracts/defi/MoonscapeDefi.sol#344-358):
        External calls:
        - token.safeTransferFrom(msg.sender,address(this),_amount) (contracts/defi/MoonscapeDefi.sol#354)
        Event emitted after the call(s):
        - TokenChangeMoondust(_sessionId,_typeId,tokenChange.tokenAddress,tokenChange.ration,_amount,msg.sender,block.timestamp) (contracts/defi/MoonscapeDefi.sol#356
)
Reentrancy in MoonscapeDefi.unStakeToken(uint256,uint256,uint256) (contracts/defi/MoonscapeDefi.sol#228-240):
        External calls:
        - withdraw(stakeKey,msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#233)
                - _rewardToken.transfer(_to,_amount) (contracts/defi/MoonscapeDefi.sol#303)
        - token.safeTransfer(msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#237)
        External calls sending eth:
        - withdraw(stakeKey,msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#233)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        Event emitted after the call(s):
        - UnStakeToken(msg.sender,_sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#239)
Reentrancy in MoonscapeBeta.unlock(uint256) (contracts/beta/MoonscapeBeta.sol#58-69):
        External calls:
        - session.token.safeTransfer(msg.sender,session.requiredAmount) (contracts/beta/MoonscapeBeta.sol#66)
        Event emitted after the call(s):
        - Transfer(msg.sender,false) (contracts/beta/MoonscapeBeta.sol#68)
Reentrancy in MoonscapeGame.unstakeForMoondust(uint256) (contracts/game/MoonscapeGame.sol#112-126):
        External calls:
        - require(bool,string)(_token.transfer(msg.sender,_amount),Failed to transfer token from contract to user) (contracts/game/MoonscapeGame.sol#121)
        Event emitted after the call(s):
        - Unstake(msg.sender,_amount,block.timestamp,_balance.stakeAmount) (contracts/game/MoonscapeGame.sol#125)
```

```
Reentrancy in MscpVesting.withdraw() (contracts/mscp/MscpVesting.sol#72-90):
        External calls:
        - token.safeTransfer(msg.sender,availableAmount) (contracts/mscp/MscpVesting.sol#87)
        Event emitted after the call(s):
        - Withdraw(msg.sender,availableAmount,balance.remainingCoins) (contracts/mscp/MscpVesting.sol#89)
Reentrancy in MscpVesting30M.withdraw() (contracts/mscp/MscpVesting30M.sol#57-81):
        External calls:
        - token.safeTransfer(msg.sender,availableAmount) (contracts/mscp/MscpVesting30M.sol#78)
        Event emitted after the call(s):
        - Withdraw(msg.sender,availableAmount,remainingCoins[msg.sender]) (contracts/mscp/MscpVesting30M.sol#80)
Reentrancy in MscpVesting5M.withdraw() (contracts/mscp/MscpVesting5M.sol#57-81):
        External calls:
        - token.safeTransfer(msg.sender,availableAmount) (contracts/mscp/MscpVesting5M.sol#78)
        Event emitted after the call(s):
        - Withdraw(msg.sender,availableAmount,remainingCoins[msg.sender]) (contracts/mscp/MscpVesting5M.sol#80)
Reentrancy in MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32) (contracts/defi/MoonscapeDefi.sol#212-224):
        External calls:
        - require(bool)(giveBonus(tokenStaking.sessionId,_stakeId,tokenStaking.rewardToken,_bonusPercent)) (contracts/defi/MoonscapeDefi.sol#218)
                - res = rewardToken.transferFrom(owner(),msg.sender,_bonusPercent) (contracts/defi/MoonscapeDefi.sol#271)
        - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
                - returndata = address(token).functionCall(data,SafeERC20: low-level call failed) (contracts/openzeppelin/contracts/token/ERC20/SafeERC20.sol#69)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
                - token.safeTransfer(msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#237)
                - _rewardToken.transfer(_to,_amount) (contracts/defi/MoonscapeDefi.sol#303)
        External calls sending eth:
        - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        Event emitted after the call(s):
        - Reward(stakerAddr,key,interest) (contracts/defi/Stake.sol#232)
                - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
        - UnStakeToken(msg.sender,_sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#239)
                - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
        - Withdraw(stakerAddr,key,amount) (contracts/defi/Stake.sol#144)
                - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
        - WithdrawAll(tokenStaking.sessionId,_stakeId,_cityId,_buildingId,_amount,_bonusPercent,msg.sender,block.timestamp) (contracts/defi/MoonscapeDefi.sol#223)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

MoonscapeBeta.startSession(IERC20,uint256,uint256,uint256) (contracts/beta/MoonscapeBeta.sol#30-41) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_startTime > now,session should start in future) (contracts/beta/MoonscapeBeta.sol#32)
MoonscapeBeta.lock(uint256) (contracts/beta/MoonscapeBeta.sol#43-56) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(session.startTime < now,session hasnt started yet) (contracts/beta/MoonscapeBeta.sol#45)
        - require(bool,string)(session.endTime > now,session is finished) (contracts/beta/MoonscapeBeta.sol#46)
MoonscapeBeta.unlock(uint256) (contracts/beta/MoonscapeBeta.sol#58-69) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool)(now > session.endTime) (contracts/beta/MoonscapeBeta.sol#60)
MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32) (contracts/defi/MoonscapeDefi.sol#212-224) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp > session.endTime (contracts/defi/MoonscapeDefi.sol#216)
MoonscapeDefi._safeTransfer(address,address,uint256) (contracts/defi/MoonscapeDefi.sol#295-313) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_amount <= _balance,do not have enough token to reward) (contracts/defi/MoonscapeDefi.sol#300)
        - require(bool,string)(_rewardToken.balanceOf(_to) == _beforBalance + _amount,Invalid transfer) (contracts/defi/MoonscapeDefi.sol#305)
        - require(bool,string)(_amount <= _balance_scope_0,do not have enough token to reward) (contracts/defi/MoonscapeDefi.sol#309)
MoonscapeDefi.validSessionTime(uint256,uint256) (contracts/defi/MoonscapeDefi.sol#373-381) uses timestamp for comparisons
        Dangerous comparisons:
        - _startTime > session.endTime && _startTime >= block.timestamp && _startTime < _endTime (contracts/defi/MoonscapeDefi.sol#376)
Stake.withdraw(bytes32,address,uint256) (contracts/defi/Stake.sol#124-145) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(amount > 0 && staker.deposit >= amount,STAKE_TOKEN: stake amount zero) (contracts/defi/Stake.sol#130)
Stake._reward(bytes32,address) (contracts/defi/Stake.sol#210-235) uses timestamp for comparisons
        Dangerous comparisons:
        - interest == 0 (contracts/defi/Stake.sol#219)
Stake.claimable(bytes32,address) (contracts/defi/Stake.sol#237-260) uses timestamp for comparisons
        Dangerous comparisons:
        - sessionCap == period.endTime && staker.rewardClaimedTime >= sessionCap (contracts/defi/Stake.sol#252)
Stake.getPeriodTime(uint256,uint256) (contracts/defi/Stake.sol#268-277) uses timestamp for comparisons
        Dangerous comparisons:
        - block.timestamp < startTime (contracts/defi/Stake.sol#270)
Stake.isActive(uint256,uint256) (contracts/defi/Stake.sol#279-285) uses timestamp for comparisons
        Dangerous comparisons:
        - (block.timestamp >= startTime && block.timestamp <= endTime) (contracts/defi/Stake.sol#284)
Stake.isActive(bytes32) (contracts/defi/Stake.sol#290-295) uses timestamp for comparisons
        Dangerous comparisons:
        - (block.timestamp >= period.startTime && block.timestamp <= period.endTime) (contracts/defi/Stake.sol#294)
Stake.initiated(bytes32) (contracts/defi/Stake.sol#300-305) uses timestamp for comparisons
        Dangerous comparisons:
        - (block.timestamp <= period.endTime) (contracts/defi/Stake.sol#304)
StakeNftForChain.isActive(uint256,uint256) (contracts/defi/StakeNftForChain.sol#78-84) uses timestamp for comparisons
        Dangerous comparisons:
        - (block.timestamp >= startTime && block.timestamp <= endTime) (contracts/defi/StakeNftForChain.sol#83)
StakeNftForChain.isActive(uint256) (contracts/defi/StakeNftForChain.sol#89-94) uses timestamp for comparisons
        Dangerous comparisons:
        - (block.timestamp >= period.startTime && block.timestamp <= period.endTime) (contracts/defi/StakeNftForChain.sol#93)
MscpVesting.constructor(IERC20,uint256) (contracts/mscp/MscpVesting.sol#36-42) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_startTime > now,vesting should start in future) (contracts/mscp/MscpVesting.sol#38)
MscpVesting.addInvestor(address,bool) (contracts/mscp/MscpVesting.sol#51-61) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(balances[_investor].remainingCoins == 0,investor already has allocation) (contracts/mscp/MscpVesting.sol#52)
```

```
        - require(bool,string)(balances[_investor].remainingCoins == 0,investor already has allocation) (contracts/mscp/MscpVesting.sol#52)
MscpVesting.disableInvestor(address) (contracts/mscp/MscpVesting.sol#65-69) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(balances[_investor].remainingCoins > 0,investor already disabled) (contracts/mscp/MscpVesting.sol#66)
MscpVesting.withdraw() (contracts/mscp/MscpVesting.sol#72-90) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(now >= startTime,vesting hasnt started yet) (contracts/mscp/MscpVesting.sol#74)
        - require(bool,string)(balance.remainingCoins > 0,user has no allocation) (contracts/mscp/MscpVesting.sol#75)
MscpVesting.getRemainingTime() (contracts/mscp/MscpVesting.sol#104-113) uses timestamp for comparisons
        Dangerous comparisons:
        - now < startTime + DURATION_STRATEGIC (contracts/mscp/MscpVesting.sol#106)
        - now < startTime + DURATION_PRIVATE (contracts/mscp/MscpVesting.sol#109)
MscpVesting.getDuration(bool) (contracts/mscp/MscpVesting.sol#123-133) uses timestamp for comparisons
        Dangerous comparisons:
        - now < startTime + DURATION_STRATEGIC (contracts/mscp/MscpVesting.sol#125)
        - now < startTime + DURATION_PRIVATE (contracts/mscp/MscpVesting.sol#129)
MscpVesting30M.constructor(IERC20,uint256) (contracts/mscp/MscpVesting30M.sol#28-34) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_startTime > now,vesting should start in future) (contracts/mscp/MscpVesting30M.sol#30)
MscpVesting30M.addInvestor(address) (contracts/mscp/MscpVesting30M.sol#42-46) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(remainingCoins[_investor] == 0,investor already has allocation) (contracts/mscp/MscpVesting30M.sol#43)
MscpVesting30M.disableInvestor(address) (contracts/mscp/MscpVesting30M.sol#50-54) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(remainingCoins[_investor] > 0,investor already disabled) (contracts/mscp/MscpVesting30M.sol#51)
MscpVesting30M.withdraw() (contracts/mscp/MscpVesting30M.sol#57-81) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(now > startTime,vesting hasnt started yet) (contracts/mscp/MscpVesting30M.sol#58)
        - require(bool,string)(remainingCoins[msg.sender] > 0,user has no allocation) (contracts/mscp/MscpVesting30M.sol#59)
        - now < startTime + DURATION (contracts/mscp/MscpVesting30M.sol#63)
MscpVesting5M.constructor(IERC20,uint256) (contracts/mscp/MscpVesting5M.sol#28-34) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_startTime > now,vesting should start in future) (contracts/mscp/MscpVesting5M.sol#30)
MscpVesting5M.addInvestor(address) (contracts/mscp/MscpVesting5M.sol#42-46) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(remainingCoins[_investor] == 0,investor already has allocation) (contracts/mscp/MscpVesting5M.sol#43)
MscpVesting5M.disableInvestor(address) (contracts/mscp/MscpVesting5M.sol#50-54) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(remainingCoins[_investor] > 0,investor already disabled) (contracts/mscp/MscpVesting5M.sol#51)
MscpVesting5M.withdraw() (contracts/mscp/MscpVesting5M.sol#57-81) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(now > startTime,vesting hasnt started yet) (contracts/mscp/MscpVesting5M.sol#58)
        - require(bool,string)(remainingCoins[msg.sender] > 0,user has no allocation) (contracts/mscp/MscpVesting5M.sol#59)
        - now < startTime + DURATION (contracts/mscp/MscpVesting5M.sol#63)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
```

```
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

MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32) (contracts/defi/MoonscapeDefi.sol#243-261) compares to a boolean constant:
        -require(bool,string)(receiveBonus[stakeKey][msg.sender] == false,already rewarded) (contracts/defi/MoonscapeDefi.sol#247)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Different versions of Solidity are used:
        - Version used: ['0.6.7', '^0.6.0', '^0.6.2', '^0.6.7']
        - 0.6.7 (contracts/Migrations.sol#1)
        - 0.6.7 (contracts/beta/MoonscapeBeta.sol#1)
        - 0.6.7 (contracts/defi/MoonscapeDefi.sol#1)
        - 0.6.7 (contracts/defi/Stake.sol#1)
        - 0.6.7 (contracts/defi/StakeNftForChain.sol#2)
        - 0.6.7 (contracts/game/MoonscapeGame.sol#1)
        - 0.6.7 (contracts/mscp/MscpToken.sol#2)
        - 0.6.7 (contracts/mscp/MscpVesting.sol#1)
        - 0.6.7 (contracts/mscp/MscpVesting30M.sol#1)
        - 0.6.7 (contracts/mscp/MscpVesting5M.sol#1)
        - 0.6.7 (contracts/nfts/CityNft.sol#3)
        - 0.6.7 (contracts/nfts/RoverNft.sol#3)
        - 0.6.7 (contracts/nfts/SeascapeNft.sol#3)
        - ^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)
        - ^0.6.7 (contracts/openzeppelin/contracts/security/ReentrancyGuard.sol#3)
        - ^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/ERC20.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
```

```
AccessControl._setRoleAdmin(bytes32,bytes32) (contracts/openzeppelin/contracts/access/AccessControl.sol#201-204) is never used and should be removed
AccessControl._setupRole(bytes32,address) (contracts/openzeppelin/contracts/access/AccessControl.sol#192-194) 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
ERC20._burn(address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#250-258) is never used and should be removed
ERC20._mint(address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#229-237) is never used and should be removed
ERC20._setupDecimals(uint8) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#288-290) 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
ReentrancyGuard.initReentrancyStatus() (contracts/openzeppelin/contracts/security/ReentrancyGuard.sol#47-49) 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
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
Stake._claim(bytes32,address,uint256) (contracts/defi/Stake.sol#208) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
```

```
Pragma version0.6.7 (contracts/Migrations.sol#1) allows old versions
Pragma version0.6.7 (contracts/beta/MoonscapeBeta.sol#1) allows old versions
Pragma version0.6.7 (contracts/defi/MoonscapeDefi.sol#1) allows old versions
Pragma version0.6.7 (contracts/defi/Stake.sol#1) allows old versions
Pragma version0.6.7 (contracts/defi/StakeNftForChain.sol#2) allows old versions
Pragma version0.6.7 (contracts/game/MoonscapeGame.sol#1) allows old versions
Pragma version0.6.7 (contracts/mscp/MscpToken.sol#2) allows old versions
Pragma version0.6.7 (contracts/mscp/MscpVesting.sol#1) allows old versions
Pragma version0.6.7 (contracts/mscp/MscpVesting30M.sol#1) allows old versions
Pragma version0.6.7 (contracts/mscp/MscpVesting5M.sol#1) allows old versions
Pragma version0.6.7 (contracts/nfts/CityNft.sol#3) allows old versions
Pragma version0.6.7 (contracts/nfts/RoverNft.sol#3) allows old versions
Pragma version0.6.7 (contracts/nfts/SeascapeNft.sol#3) 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^0.6.7 (contracts/openzeppelin/contracts/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.6.0 (contracts/openzeppelin/contracts/token/ERC20/ERC20.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 Migrations.upgrade(address).new_address (contracts/Migrations.sol#19) is not in mixedCase
Variable Migrations.last_completed_migration (contracts/Migrations.sol#5) is not in mixedCase
Parameter MoonscapeBeta.startSession(IERC20,uint256,uint256,uint256)._token (contracts/beta/MoonscapeBeta.sol#30) is not in mixedCase
Parameter MoonscapeBeta.startSession(IERC20,uint256,uint256,uint256)._requiredAmount (contracts/beta/MoonscapeBeta.sol#30) is not in mixedCase
Parameter MoonscapeBeta.startSession(IERC20,uint256,uint256,uint256)._startTime (contracts/beta/MoonscapeBeta.sol#30) is not in mixedCase
Parameter MoonscapeBeta.startSession(IERC20,uint256,uint256,uint256)._endTime (contracts/beta/MoonscapeBeta.sol#30) is not in mixedCase
Parameter MoonscapeBeta.lock(uint256)._sessionId (contracts/beta/MoonscapeBeta.sol#43) is not in mixedCase
Parameter MoonscapeBeta.unlock(uint256)._sessionId (contracts/beta/MoonscapeBeta.sol#58) is not in mixedCase
Parameter MoonscapeDefi.startSession(uint256,uint256,address)._startTime (contracts/defi/MoonscapeDefi.sol#70) is not in mixedCase
Parameter MoonscapeDefi.startSession(uint256,uint256,address)._endTime (contracts/defi/MoonscapeDefi.sol#70) is not in mixedCase
Parameter MoonscapeDefi.startSession(uint256,uint256,address)._verifier (contracts/defi/MoonscapeDefi.sol#70) is not in mixedCase
Parameter MoonscapeDefi.pauseSession(uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#83) is not in mixedCase
Parameter MoonscapeDefi.resumeSession(uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#94) is not in mixedCase
Parameter MoonscapeDefi.addTokenStaking(uint256,address,uint256,address,bool)._sessionId (contracts/defi/MoonscapeDefi.sol#105) is not in mixedCase
Parameter MoonscapeDefi.addTokenStaking(uint256,address,uint256,address,bool)._burn (contracts/defi/MoonscapeDefi.sol#105) is not in mixedCase
Parameter MoonscapeDefi.stakeToken(uint256,uint256,uint256,uint256,uint8,bytes32[2])._stakeId (contracts/defi/MoonscapeDefi.sol#132) is not in mixedCase
Parameter MoonscapeDefi.stakeToken(uint256,uint256,uint256,uint256,uint8,bytes32[2])._cityId (contracts/defi/MoonscapeDefi.sol#132) is not in mixedCase
Parameter MoonscapeDefi.stakeToken(uint256,uint256,uint256,uint256,uint8,bytes32[2])._buildingId (contracts/defi/MoonscapeDefi.sol#132) is not in mixedCase
Parameter MoonscapeDefi.stakeToken(uint256,uint256,uint256,uint256,uint8,bytes32[2])._amount (contracts/defi/MoonscapeDefi.sol#132) is not in mixedCase
Parameter MoonscapeDefi.claim(uint256,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#168) is not in mixedCase
Parameter MoonscapeDefi.claim(uint256,uint256)._stakeId (contracts/defi/MoonscapeDefi.sol#168) is not in mixedCase
Parameter MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._stakeId (contracts/defi/MoonscapeDefi.sol#180) is not in mixedCase
Parameter MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._cityId (contracts/defi/MoonscapeDefi.sol#180) is not in mixedCase
Parameter MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._buildingId (contracts/defi/MoonscapeDefi.sol#180) is not in mixedCase
Parameter MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._scapeNftId (contracts/defi/MoonscapeDefi.sol#180) is not in mixedCase
Parameter MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._v (contracts/defi/MoonscapeDefi.sol#180) is not in mixedCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._stakeId (contracts/defi/MoonscapeDefi.sol#212) is not in mixedCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._cityId (contracts/defi/MoonscapeDefi.sol#212) is not in mixedCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._buildingId (contracts/defi/MoonscapeDefi.sol#212) is not in mixedC
ase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._amount (contracts/defi/MoonscapeDefi.sol#212) is not in mixedCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._bonusPercent (contracts/defi/MoonscapeDefi.sol#212) is not in mixe
dCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._v (contracts/defi/MoonscapeDefi.sol#212) is not in mixedCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._r (contracts/defi/MoonscapeDefi.sol#212) is not in mixedCase
Parameter MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._s (contracts/defi/MoonscapeDefi.sol#212) is not in mixedCase
Parameter MoonscapeDefi.unStakeToken(uint256,uint256,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#228) is not in mixedCase
Parameter MoonscapeDefi.unStakeToken(uint256,uint256,uint256)._stakeId (contracts/defi/MoonscapeDefi.sol#228) is not in mixedCase
Parameter MoonscapeDefi.unStakeToken(uint256,uint256,uint256)._amount (contracts/defi/MoonscapeDefi.sol#228) is not in mixedCase
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._sessionId (contracts/defi/MoonscapeDefi.sol#243) is not in mixedCa
se
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._stakeId (contracts/defi/MoonscapeDefi.sol#243) is not in mixedCase
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._cityId (contracts/defi/MoonscapeDefi.sol#243) is not in mixedCase
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._buildingId (contracts/defi/MoonscapeDefi.sol#243) is not in mixedC
ase
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._bonusPercent (contracts/defi/MoonscapeDefi.sol#243) is not in mixe
dCase
```

```
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._v (contracts/defi/MoonscapeDefi.sol#243) is not in mixedCase
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._r (contracts/defi/MoonscapeDefi.sol#243) is not in mixedCase
Parameter MoonscapeDefi.verifyBonus(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32)._s (contracts/defi/MoonscapeDefi.sol#243) is not in mixedCase
Parameter MoonscapeDefi.giveBonus(uint256,uint256,address,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#265) is not in mixedCase
Parameter MoonscapeDefi.giveBonus(uint256,uint256,address,uint256)._stakeId (contracts/defi/MoonscapeDefi.sol#265) is not in mixedCase
Parameter MoonscapeDefi.giveBonus(uint256,uint256,address,uint256)._rewardToken (contracts/defi/MoonscapeDefi.sol#265) is not in mixedCase
Parameter MoonscapeDefi.giveBonus(uint256,uint256,address,uint256)._bonusPercent (contracts/defi/MoonscapeDefi.sol#265) is not in mixedCase
Parameter MoonscapeDefi.addTokenChangeMoondustType(uint256,uint256,address,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#317) is not in mixedCase
Parameter MoonscapeDefi.addTokenChangeMoondustType(uint256,uint256,address,uint256)._typeId (contracts/defi/MoonscapeDefi.sol#317) is not in mixedCase
Parameter MoonscapeDefi.addTokenChangeMoondustType(uint256,uint256,address,uint256)._token (contracts/defi/MoonscapeDefi.sol#317) is not in mixedCase
Parameter MoonscapeDefi.addTokenChangeMoondustType(uint256,uint256,address,uint256)._ration (contracts/defi/MoonscapeDefi.sol#317) is not in mixedCase
Parameter MoonscapeDefi.setRatio(uint256,uint256,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#331) is not in mixedCase
Parameter MoonscapeDefi.setRatio(uint256,uint256,uint256)._typeId (contracts/defi/MoonscapeDefi.sol#331) is not in mixedCase
Parameter MoonscapeDefi.setRatio(uint256,uint256,uint256)._ration (contracts/defi/MoonscapeDefi.sol#331) is not in mixedCase
Parameter MoonscapeDefi.tokenChangeMoondust(uint256,uint256,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#344) is not in mixedCase
Parameter MoonscapeDefi.tokenChangeMoondust(uint256,uint256,uint256)._typeId (contracts/defi/MoonscapeDefi.sol#344) is not in mixedCase
Parameter MoonscapeDefi.tokenChangeMoondust(uint256,uint256,uint256)._amount (contracts/defi/MoonscapeDefi.sol#344) is not in mixedCase
Parameter MoonscapeDefi.stakeKeyOf(uint256,uint256)._sessionId (contracts/defi/MoonscapeDefi.sol#367) is not in mixedCase
Parameter MoonscapeDefi.stakeKeyOf(uint256,uint256)._stakeId (contracts/defi/MoonscapeDefi.sol#367) is not in mixedCase
Parameter MoonscapeDefi.validSessionTime(uint256,uint256)._startTime (contracts/defi/MoonscapeDefi.sol#373) is not in mixedCase
Parameter MoonscapeDefi.validSessionTime(uint256,uint256)._endTime (contracts/defi/MoonscapeDefi.sol#373) is not in mixedCase
Constant MoonscapeDefi.scaler (contracts/defi/MoonscapeDefi.sol#16) is not in UPPER_CASE_WITH_UNDERSCORES
Constant MoonscapeDefi.dead (contracts/defi/MoonscapeDefi.sol#18) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter StakeNftForChain.startSession(uint256,uint256,address)._startTime (contracts/defi/StakeNftForChain.sol#37) is not in mixedCase
Parameter StakeNftForChain.startSession(uint256,uint256,address)._endTime (contracts/defi/StakeNftForChain.sol#37) is not in mixedCase
Parameter StakeNftForChain.startSession(uint256,uint256,address)._verifier (contracts/defi/StakeNftForChain.sol#37) is not in mixedCase
Parameter StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._stakeId (contracts/defi/StakeNftForChain.sol#50) is not in mixedCase
Parameter StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._cityId (contracts/defi/StakeNftForChain.sol#50) is not in mixedCase
Parameter StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._buildingId (contracts/defi/StakeNftForChain.sol#50) is not in mixedCase
Parameter StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._scapeNftId (contracts/defi/StakeNftForChain.sol#50) is not in mixedCase
Parameter StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2])._v (contracts/defi/StakeNftForChain.sol#50) is not in mixedCase
Parameter StakeNftForChain.isActive(uint256)._sessionId (contracts/defi/StakeNftForChain.sol#89) is not in mixedCase
Parameter MoonscapeGame.purchaseMoondust(uint256)._amount (contracts/game/MoonscapeGame.sol#86) is not in mixedCase
Parameter MoonscapeGame.stakeForMoondust(uint256)._amount (contracts/game/MoonscapeGame.sol#99) is not in mixedCase
Parameter MoonscapeGame.unstakeForMoondust(uint256)._amount (contracts/game/MoonscapeGame.sol#112) is not in mixedCase
Parameter MoonscapeGame.importCity(uint256)._id (contracts/game/MoonscapeGame.sol#135) is not in mixedCase
Parameter MoonscapeGame.exportCity(uint256)._id (contracts/game/MoonscapeGame.sol#147) is not in mixedCase
Parameter MoonscapeGame.mintCity(uint256,uint8,uint8,bytes32,bytes32)._id (contracts/game/MoonscapeGame.sol#158) is not in mixedCase
Parameter MoonscapeGame.mintCity(uint256,uint8,uint8,bytes32,bytes32)._category (contracts/game/MoonscapeGame.sol#158) is not in mixedCase
Parameter MoonscapeGame.mintCity(uint256,uint8,uint8,bytes32,bytes32)._v (contracts/game/MoonscapeGame.sol#158) is not in mixedCase
Parameter MoonscapeGame.mintCity(uint256,uint8,uint8,bytes32,bytes32)._r (contracts/game/MoonscapeGame.sol#158) is not in mixedCase
Parameter MoonscapeGame.mintCity(uint256,uint8,uint8,bytes32,bytes32)._s (contracts/game/MoonscapeGame.sol#158) is not in mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._sessionId (contracts/game/MoonscapeGame.sol#181) is no
t in mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._stakeId (contracts/game/MoonscapeGame.sol#181) is not
in mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._cityId (contracts/game/MoonscapeGame.sol#181) is not i
n mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._buildingId (contracts/game/MoonscapeGame.sol#181) is n
ot in mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._scapeNftId (contracts/game/MoonscapeGame.sol#181) is n
ot in mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._power (contracts/game/MoonscapeGame.sol#181) is not in
 mixedCase
Parameter MoonscapeGame.burnScapeForBuilding(uint256,uint256,uint256,uint256,uint256,uint256,uint8,bytes32[2])._v (contracts/game/MoonscapeGame.sol#181) is not in mix
edCase
Parameter MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32)._sessionId (contracts/game/MoonscapeGame.sol#204) is not in mixedCase
Parameter MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32)._scapeNftId (contracts/game/MoonscapeGame.sol#204) is not in mixedCase
Parameter MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32)._v (contracts/game/MoonscapeGame.sol#204) is not in mixedCase
Parameter MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32)._r (contracts/game/MoonscapeGame.sol#204) is not in mixedCase
Parameter MoonscapeGame.burnScapeForConnection(uint256,uint256,uint8,bytes32,bytes32)._s (contracts/game/MoonscapeGame.sol#204) is not in mixedCase
Parameter MoonscapeGame.importRover(uint256)._id (contracts/game/MoonscapeGame.sol#232) is not in mixedCase
Parameter MoonscapeGame.exportRover(uint256)._id (contracts/game/MoonscapeGame.sol#244) is not in mixedCase
Parameter MoonscapeGame.mintRover(uint256,uint8,uint8,bytes32,bytes32)._id (contracts/game/MoonscapeGame.sol#255) is not in mixedCase
Parameter MoonscapeGame.mintRover(uint256,uint8,uint8,bytes32,bytes32)._type (contracts/game/MoonscapeGame.sol#255) is not in mixedCase
Parameter MoonscapeGame.mintRover(uint256,uint8,uint8,bytes32,bytes32)._v (contracts/game/MoonscapeGame.sol#255) is not in mixedCase
Parameter MoonscapeGame.mintRover(uint256,uint8,uint8,bytes32,bytes32)._r (contracts/game/MoonscapeGame.sol#255) is not in mixedCase
Parameter MoonscapeGame.mintRover(uint256,uint8,uint8,bytes32,bytes32)._s (contracts/game/MoonscapeGame.sol#255) is not in mixedCase
Constant MoonscapeGame.scaler (contracts/game/MoonscapeGame.sol#15) is not in UPPER_CASE_WITH_UNDERSCORES
Variable MoonscapeGame.MSCP (contracts/game/MoonscapeGame.sol#17) is not in mixedCase
Constant MoonscapeGame.dead (contracts/game/MoonscapeGame.sol#22) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter MscpToken.addBridge(address)._bridge (contracts/mscp/MscpToken.sol#56) is not in mixedCase
Parameter MscpToken.removeBridge(address)._bridge (contracts/mscp/MscpToken.sol#65) is not in mixedCase
Constant MscpToken._name (contracts/mscp/MscpToken.sol#23) is not in UPPER_CASE_WITH_UNDERSCORES
Constant MscpToken._symbol (contracts/mscp/MscpToken.sol#24) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter MscpVesting.addInvestor(address,bool)._investor (contracts/mscp/MscpVesting.sol#51) is not in mixedCase
Parameter MscpVesting.addInvestor(address,bool)._strategicInvestor (contracts/mscp/MscpVesting.sol#51) is not in mixedCase
Parameter MscpVesting.disableInvestor(address)._investor (contracts/mscp/MscpVesting.sol#65) is not in mixedCase
Parameter MscpVesting.getDuration(bool)._strategicInvestor (contracts/mscp/MscpVesting.sol#123) is not in mixedCase
Parameter MscpVesting.getAvailableTokens(bool,uint256,uint256)._strategicInvestor (contracts/mscp/MscpVesting.sol#141) is not in mixedCase
Parameter MscpVesting.getAvailableTokens(bool,uint256,uint256)._timePassed (contracts/mscp/MscpVesting.sol#142) is not in mixedCase
Parameter MscpVesting.getAvailableTokens(bool,uint256,uint256)._remainingCoins (contracts/mscp/MscpVesting.sol#143) is not in mixedCase
Parameter MscpVesting.getBonus(bool)._strategicInvestor (contracts/mscp/MscpVesting.sol#161) is not in mixedCase
Parameter MscpVesting30M.addInvestor(address)._investor (contracts/mscp/MscpVesting30M.sol#42) is not in mixedCase
Parameter MscpVesting30M.disableInvestor(address)._investor (contracts/mscp/MscpVesting30M.sol#50) is not in mixedCase
Parameter MscpVesting5M.addInvestor(address)._investor (contracts/mscp/MscpVesting5M.sol#42) is not in mixedCase
Parameter MscpVesting5M.disableInvestor(address)._investor (contracts/mscp/MscpVesting5M.sol#50) is not in mixedCase
Parameter CityNft.mint(uint256,uint8,address)._tokenId (contracts/nfts/CityNft.sol#24) is not in mixedCase
Parameter CityNft.mint(uint256,uint8,address)._category (contracts/nfts/CityNft.sol#24) is not in mixedCase
Parameter CityNft.mint(uint256,uint8,address)._to (contracts/nfts/CityNft.sol#24) is not in mixedCase
Parameter CityNft.setMinter(address)._minter (contracts/nfts/CityNft.sol#37) is not in mixedCase
Parameter CityNft.unsetMinter(address)._minter (contracts/nfts/CityNft.sol#45) is not in mixedCase
```

AUTOMATED TESTING

```
Parameter CityNft.setBaseUri(string)._uri (contracts/nfts/CityNft.sol#53) is not in mixedCase
Parameter RoverNft.mint(uint256,uint8,address)._tokenId (contracts/nfts/RoverNft.sol#24) is not in mixedCase
Parameter RoverNft.mint(uint256,uint8,address)._type (contracts/nfts/RoverNft.sol#24) is not in mixedCase
Parameter RoverNft.mint(uint256,uint8,address)._to (contracts/nfts/RoverNft.sol#24) is not in mixedCase
Parameter RoverNft.setMinter(address)._minter (contracts/nfts/RoverNft.sol#37) is not in mixedCase
Parameter RoverNft.unsetMinter(address)._minter (contracts/nfts/RoverNft.sol#45) is not in mixedCase
Parameter RoverNft.setBaseUri(string)._uri (contracts/nfts/RoverNft.sol#53) is not in mixedCase
Parameter SeascapeNft.mint(address,uint256,uint8)._to (contracts/nfts/SeascapeNft.sol#48) is not in mixedCase
Parameter SeascapeNft.mint(address,uint256,uint8)._generation (contracts/nfts/SeascapeNft.sol#48) is not in mixedCase
Parameter SeascapeNft.mint(address,uint256,uint8)._quality (contracts/nfts/SeascapeNft.sol#48) is not in mixedCase
Parameter SeascapeNft.setOwner(address)._owner (contracts/nfts/SeascapeNft.sol#61) is not in mixedCase
Parameter SeascapeNft.setFactory(address)._factory (contracts/nfts/SeascapeNft.sol#65) is not in mixedCase
Parameter SeascapeNft.setBaseUri(string)._uri (contracts/nfts/SeascapeNft.sol#69) 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 MoonscapeDefi.stakeToken(uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/defi/MoonscapeDefi.sol#132-164):
        External calls:
        - deposit(stakeKey,msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#155)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        Event emitted after the call(s):
        - StakeToken(msg.sender,tokenStaking.sessionId,_stakeId,_cityId,_buildingId,_amount,nonce) (contracts/defi/MoonscapeDefi.sol#163)
Reentrancy in MoonscapeDefi.unStakeToken(uint256,uint256,uint256) (contracts/defi/MoonscapeDefi.sol#228-240):
        External calls:
        - withdraw(stakeKey,msg.sender,_amount) (contracts/defi/MoonscapeDefi.sol#233)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        Event emitted after the call(s):
        - UnStakeToken(msg.sender,_sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#239)
Reentrancy in MoonscapeDefi.withdrawAll(uint256,uint256,uint256,uint256,uint256,uint8,bytes32,bytes32) (contracts/defi/MoonscapeDefi.sol#212-224):
        External calls:
        - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        External calls sending eth:
        - unStakeToken(tokenStaking.sessionId,_stakeId,_amount) (contracts/defi/MoonscapeDefi.sol#221)
                - (success,returndata) = target.call{value: weiValue}(data) (contracts/openzeppelin/contracts/utils/Address.sol#123)
                - address(_to).transfer(_amount) (contracts/defi/MoonscapeDefi.sol#311)
        Event emitted after the call(s):
        - WithdrawAll(tokenStaking.sessionId,_stakeId,_cityId,_buildingId,_amount,_bonusPercent,msg.sender,block.timestamp) (contracts/defi/MoonscapeDefi.sol#223)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

MoonscapeDefi.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/defi/MoonscapeDefi.sol#180-208) uses literals with too many digits:
        - nft.safeTransferFrom(msg.sender,0x000000000000000000000000000000000000dEaD,_scapeNftId) (contracts/defi/MoonscapeDefi.sol#204)
MoonscapeDefi.slitherConstructorConstantVariables() (contracts/defi/MoonscapeDefi.sol#11-405) uses literals with too many digits:
        - dead = 0x000000000000000000000000000000000000dEaD (contracts/defi/MoonscapeDefi.sol#18)
StakeNftForChain.importNft(uint256,uint256,uint256,uint256,uint8,bytes32[2]) (contracts/defi/StakeNftForChain.sol#50-70) uses literals with too many digits:
        - nft.safeTransferFrom(msg.sender,0x000000000000000000000000000000000000dEaD,_scapeNftId) (contracts/defi/StakeNftForChain.sol#67)
MoonscapeGame.slitherConstructorConstantVariables() (contracts/game/MoonscapeGame.sol#11-286) uses literals with too many digits:
        - dead = 0x000000000000000000000000000000000000dEaD (contracts/game/MoonscapeGame.sol#22)
MscpToken.slitherConstructorVariables() (contracts/mscp/MscpToken.sol#14-386) uses literals with too many digits:
        - limitSupply = 111111111000000000000000000 (contracts/mscp/MscpToken.sol#30)
MscpVesting.getBonus(bool) (contracts/mscp/MscpVesting.sol#161-165) uses literals with too many digits:
        - 2000000 * 10 ** 18 (contracts/mscp/MscpVesting.sol#163)
MscpVesting.getBonus(bool) (contracts/mscp/MscpVesting.sol#161-165) uses literals with too many digits:
        - 1500000 * 10 ** 18 (contracts/mscp/MscpVesting.sol#164)
MscpVesting.slitherConstructorConstantVariables() (contracts/mscp/MscpVesting.sol#11-166) uses literals with too many digits:
        - SUPPLY_PRIVATE = 8500000 * 10 ** 18 (contracts/mscp/MscpVesting.sol#19)
MscpVesting.slitherConstructorConstantVariables() (contracts/mscp/MscpVesting.sol#11-166) uses literals with too many digits:
        - SUPPLY_STRATEGIC = 8000000 * 10 ** 18 (contracts/mscp/MscpVesting.sol#20)
MscpVesting30M.withdraw() (contracts/mscp/MscpVesting30M.sol#57-81) uses literals with too many digits:
        - availableAmount += 4500000 * 10 ** 18 (contracts/mscp/MscpVesting30M.sol#75)
MscpVesting30M.slitherConstructorConstantVariables() (contracts/mscp/MscpVesting30M.sol#10-82) uses literals with too many digits:
        - SUPPLY = 25500000 * 10 ** 18 (contracts/mscp/MscpVesting30M.sol#18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

MoonscapeDefi.scaler (contracts/defi/MoonscapeDefi.sol#16) is never used in MoonscapeDefi (contracts/defi/MoonscapeDefi.sol#11-405)
MoonscapeDefi.dead (contracts/defi/MoonscapeDefi.sol#18) is never used in MoonscapeDefi (contracts/defi/MoonscapeDefi.sol#11-405)
MoonscapeGame.scaler (contracts/game/MoonscapeGame.sol#15) is never used in MoonscapeGame (contracts/game/MoonscapeGame.sol#11-286)
MscpToken.SCALER (contracts/mscp/MscpToken.sol#27) is never used in MscpToken (contracts/mscp/MscpToken.sol#14-386)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

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

```
setCompleted(uint256) should be declared external:
        - Migrations.setCompleted(uint256) (contracts/Migrations.sol#15-17)
upgrade(address) should be declared external:
        - Migrations.upgrade(address) (contracts/Migrations.sol#19-22)
addTokenChangeMoondustType(uint256,uint256,address,uint256) should be declared external:
        - MoonscapeDefi.addTokenChangeMoondustType(uint256,uint256,address,uint256) (contracts/defi/MoonscapeDefi.sol#317-327)
setRatio(uint256,uint256,uint256) should be declared external:
        - MoonscapeDefi.setRatio(uint256,uint256,uint256) (contracts/defi/MoonscapeDefi.sol#331-340)
tokenChangeMoondust(uint256,uint256,uint256) should be declared external:
        - MoonscapeDefi.tokenChangeMoondust(uint256,uint256,uint256) (contracts/defi/MoonscapeDefi.sol#344-358)
initiated(bytes32) should be declared external:
        - Stake.initiated(bytes32) (contracts/defi/Stake.sol#300-305)
burn(uint256) should be declared external:
        - MscpToken.burn(uint256) (contracts/mscp/MscpToken.sol#95-97)
burnFrom(address,uint256) should be declared external:
        - MscpToken.burnFrom(address,uint256) (contracts/mscp/MscpToken.sol#110-117)
name() should be declared external:
        - MscpToken.name() (contracts/mscp/MscpToken.sol#126-128)
symbol() should be declared external:
        - MscpToken.symbol() (contracts/mscp/MscpToken.sol#134-136)
decimals() should be declared external:
        - MscpToken.decimals() (contracts/mscp/MscpToken.sol#147-149)
totalSupply() should be declared external:
        - ERC20.totalSupply() (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#98-100)
        - MscpToken.totalSupply() (contracts/mscp/MscpToken.sol#154-156)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#105-107)
        - MscpToken.balanceOf(address) (contracts/mscp/MscpToken.sol#161-163)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#117-120)
        - MscpToken.transfer(address,uint256) (contracts/mscp/MscpToken.sol#172-175)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#136-139)
        - MscpToken.approve(address,uint256) (contracts/mscp/MscpToken.sol#202-205)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#153-157)
        - MscpToken.transferFrom(address,address,uint256) (contracts/mscp/MscpToken.sol#216-226)
increaseAllowance(address,uint256) should be declared external:
        - MscpToken.increaseAllowance(address,uint256) (contracts/mscp/MscpToken.sol#240-243)
decreaseAllowance(address,uint256) should be declared external:
        - MscpToken.decreaseAllowance(address,uint256) (contracts/mscp/MscpToken.sol#259-267)
getAllocation() should be declared external:
        - MscpVesting.getAllocation() (contracts/mscp/MscpVesting.sol#98-100)
getRemainingTime() should be declared external:
        - MscpVesting.getRemainingTime() (contracts/mscp/MscpVesting.sol#104-113)
setMinter(address) should be declared external:
        - CityNft.setMinter(address) (contracts/nfts/CityNft.sol#37-43)
unsetMinter(address) should be declared external:
        - CityNft.unsetMinter(address) (contracts/nfts/CityNft.sol#45-51)
setBaseUri(string) should be declared external:
        - CityNft.setBaseUri(string) (contracts/nfts/CityNft.sol#53-55)
setMinter(address) should be declared external:
        - RoverNft.setMinter(address) (contracts/nfts/RoverNft.sol#37-43)
unsetMinter(address) should be declared external:
        - RoverNft.unsetMinter(address) (contracts/nfts/RoverNft.sol#45-51)
setBaseUri(string) should be declared external:
        - RoverNft.setBaseUri(string) (contracts/nfts/RoverNft.sol#53-55)
mint(address,uint256,uint8) should be declared external:
        - SeascapeNft.mint(address,uint256,uint8) (contracts/nfts/SeascapeNft.sol#48-59)
setOwner(address) should be declared external:
        - SeascapeNft.setOwner(address) (contracts/nfts/SeascapeNft.sol#61-63)
setFactory(address) should be declared external:
        - SeascapeNft.setFactory(address) (contracts/nfts/SeascapeNft.sol#65-67)
setBaseUri(string) should be declared external:
        - SeascapeNft.setBaseUri(string) (contracts/nfts/SeascapeNft.sol#69-71)
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)
grantRole(bytes32,address) should be declared external:
        - AccessControl.grantRole(bytes32,address) (contracts/openzeppelin/contracts/access/AccessControl.sol#135-139)
revokeRole(bytes32,address) should be declared external:
        - AccessControl.revokeRole(bytes32,address) (contracts/openzeppelin/contracts/access/AccessControl.sol#150-154)
renounceRole(bytes32,address) should be declared external:
        - AccessControl.renounceRole(bytes32,address) (contracts/openzeppelin/contracts/access/AccessControl.sol#170-174)
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)
name() should be declared external:
        - ERC20.name() (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#66-68)
symbol() should be declared external:
        - ERC20.symbol() (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#74-76)
decimals() should be declared external:
        - ERC20.decimals() (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#91-93)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#171-174)
decreaseAllowance(address,uint256) should be declared external:
        - ERC20.decreaseAllowance(address,uint256) (contracts/openzeppelin/contracts/token/ERC20/ERC20.sol#190-193)
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)
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
. analyzed (34 contracts with 78 detectors), 373 result(s) found
```

AUTOMATED TESTING

- No major issues found by Slither.

4.2 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues and to identify low-hanging fruits on the targets for this engagement. Among the tools used was MythX, a security analysis service for Ethereum smart contracts. MythX performed a scan on the smart contracts and sent the compiled results to the analyzers in order to locate any vulnerabilities.

MythX results:

Report for beta/MoonscapeBeta.sol
https://dashboard.mythx.io/#/console/analyses/75f14c08-e2c5-4047-af2e-03df0c8c7dd1

Line	SWC Title	Severity	Short Description
32	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
45	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
46	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
60	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.

Report for defi/StakeNftForChain.sol
https://dashboard.mythx.io/#/console/analyses/3b975ecf-abaa-4496-98f3-dba9939ba3f6

Line	SWC Title	Severity	Short Description
40	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
60	(SWC-110) Assert Violation	Unknown	Out of bounds array access
65	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered

Report for defi/MoonscapeDefi.sol
https://dashboard.mythx.io/#/console/analyses/a28817eb-cf28-44ec-8cd3-57f563027f59

Line	SWC Title	Severity	Short Description
16	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
74	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
113	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
148	(SWC-110) Assert Violation	Unknown	Out of bounds array access
153	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
197	(SWC-110) Assert Violation	Unknown	Out of bounds array access
202	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
258	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
305	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered

AUTOMATED TESTING

Report for defi/Stake.sol
https://dashboard.mythx.io/#/console/analyses/a28817eb-cf28-44ec-8cd3-57f563027f59

Line	SWC Title	Severity	Short Description
10	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
87	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
87	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
87	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
113	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
114	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
136	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
138	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered
184	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
189	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
189	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
189	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
205	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
205	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
256	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
257	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
257	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
259	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
259	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
259	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

Report for contracts/defi/Stake.sol
https://dashboard.mythx.io/#/console/analyses/a137a303-d8a5-435e-9440-b43ea4e13a2b

Line	SWC Title	Severity	Short Description
294	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.

Report for game/MoonscapeGame.sol
https://dashboard.mythx.io/#/console/analyses/207eb13b-cad1-4862-a231-8deff736be92

Line	SWC Title	Severity	Short Description
17	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
18	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
19	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.
20	(SWC-108) State Variable Default Visibility	Low	State variable visibility is not set.

AUTOMATED TESTING

Report for mscp/MscpVesting.sol
https://dashboard.mythx.io/#/console/analyses/c45776d9-927b-44e2-a5ab-cfdb0210dd7a

Line	SWC Title	Severity	Short Description
7	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
19	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
19	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
20	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
20	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
84	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
106	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
107	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
107	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
109	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
110	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
110	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
125	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
126	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
129	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
130	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
150	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
150	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
151	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
153	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
153	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
154	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
163	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
163	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
164	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
164	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

Report for mscp/MscpVesting30M.sol
https://dashboard.mythx.io/#/console/analyses/a1817b4b-bf06-4dda-9a7b-af98f9980448

Line	SWC Title	Severity	Short Description
7	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
18	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
18	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
64	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
70	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

Report for mscp/MscpVesting5M.sol
https://dashboard.mythx.io/#/console/analyses/cd39142c-b790-47d7-8585-d9302c1f4be1

Line	SWC Title	Severity	Short Description
7	(SWC-116) Timestamp Dependence	Low	A control flow decision is made based on The block.timestamp environment variable.
18	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
18	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
63	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+" discovered
64	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "/" discovered
69	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered
70	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "**" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "+=" discovered
75	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "*" discovered

AUTOMATED TESTING

- No major issues found by MythX.

THANK YOU FOR CHOOSING

// HALBORN