BlockLords

Audit Report

Tue May 28 2024

contact@bitslab.xyz https://twitter.com/scalebit_

ScaleBit

BlockLords Audit Report

1 Executive Summary

1.1 Project Information

Description	Blockchain Games
Type	Game
Auditors	ScaleBit
Timeline	Thu May 16 2024 - Mon May 27 2024
Languages	Solidity
Platform	Ethereum
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/blocklords/dynasty-smart-contract
Commits	b3f206f003f8ea443b9d13be22446601002babdc 59efdf7a292e22841dda677f106f27cadd033f99 7a4932ad7e57ba1393f7cf77452795b2b43068f2 5fcfbdea62fd0ccaae0277259fb742f5a4198264 35a7c5d36c3ecd531ee366c6831155f69991547f 1872e8ae129ecfcdfca0165d5685bda872018320

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
HNFT	contracts/nfts/HouseNFT.sol	c947de661d6bf4c014d7ef797dbe896817a9a2f8
BNFT	contracts/nfts/BannerNFT.sol	27efde8ab6dcd71fd0b6e629893668886eee8f61
HNFT1	contracts/nfts/HeroNFT.sol	fe0ea0173dadea839f73e5e528918fea700bfbf2
ONFT	contracts/nfts/OrbNFT.sol	eab42d1113bc3e3b66350d3bc0a92ae6c8502567
NFA	contracts/nfts/NftFactory.sol	775210b852505dbff81f5d033758178d14ed5ccc
MAR	contracts/marketplace/Marketplace.sol	f77be62606e711309d0b133e868b9927967fd707
CHE	contracts/game/Chest.sol	c0048c0d7db58937294888e2186b3661c7677f30
MIS	contracts/game/Missions.sol	4d328246a030d1f7a7e00add4ad0e53fae3c9605
NFTIH	contracts/game/NFTImportHub.sol	4701200949878e14bd89ef8bb32199445902aa43
DUE	contracts/game/Duel.sol	7f4ef45d4a312f58080aa8a26e2f1972c0b960de

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	10	9	1
Informational	0	0	0
Minor	5	5	0
Medium	1	1	0
Major	4	3	1
Critical	0	0	0

1.4 ScaleBit Audit Breakdown

ScaleBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Unchecked External Call
- Unchecked CALL Return Values
- Functionality Checks
- Reentrancy
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic issues
- Gas usage
- Fallback function usage
- tx.origin authentication
- Replay attacks
- Coding style issues

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by BlockLords to identify any potential issues and vulnerabilities in the source code of the BlockLords smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 10 issues of varying severity, listed below.

ID	Title	Severity	Status
CHE-1	Array Out-of-bounds	Major	Fixed
CHE-2	`burn` Function Cannot Be Called	Major	Fixed
CHE-3	Signature Nonce Is Not Increased	Minor	Fixed
CHE-4	`pause` Is Not Implemented	Minor	Fixed
CHE-5	`OrbNFT` Missing Permissions	Minor	Fixed
DUE-1	NFT Missing Length Check	Minor	Fixed
NFA-1	Centralization Risk	Major	Acknowledged
NFT-1	Data Truncation	Major	Fixed
NFT-2	Calldata Missing Check	Medium	Fixed
ONF-1	Incorrect Error Code	Minor	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the BlockLords Smart Contract :

BlockLords/dynasty-smart-contract/OrbNFT.sol
Owner

- Owner can set `baseUri` through `setBaseURI` function.
- Owner can set the address of `factory` through `setFactory` function.
- Owner can set the address of `verifier` through `setVerifier` function.
- Owner can transfer the Owner permissions of the contract through the `transferOwnership` function.
- Owner can give up the Owner permission of the contract through the `renounceOwnership` function.

User

- User can mint OrbNFT through `safeMint` function and signature.
- User can be used to authorize OrbNFT through the `approve` function.
- User can transfer OrbNFT through the `transferFrom` function.
- User can use the `burn` function to burn OrbNFT.
- User can transfer OrbNFT through the `safeTransferFrom` function.
- User can be used to authorize all OrbNFTs through the `setApprovalForAll` function.

Factory

- Factory can be used to mint OrbNFT of any quality through the `mint` function, without signing.

BlockLords/dynasty-smart-contract/NftFactory.sol
Deployers

- Deployers can initialize the contract Owner, `heroNft` , `bannerNft` , `orbNft` through the `constructor` function, and set the `initialOwner` to `DEFAULT_ADMIN_ROLE` .

HeroGenerator

- HeroGenerator can call `heroNft` contract mint heroNft through `mintHero` function.

BannerGenerator

- BannerGenerator can call `bannerNft` contract mint bannerNft through `mintBanner` function.

OrbGenerator

- OrbGenerator can call `orbNft` contract mint orbNft through `mintOrb` function.

Owner

- Owner can add `DEFAULT_ADMIN_ROLE` to the specified account through the `addAdmin` function.
- Owner can remove `DEFAULT_ADMIN_ROLE` from the specified account through the `removeAdmin` function.

Admin

- Admin can set the `orbNft` contract through the `setOrbNft` function.
- Admin can set the `heroNft` contract through the `setHeroNft` function.
- Admin can set the `bannerNft` contract through the `setBannerNft` function.
- Admin can add `HERO_GENERATOR_ROLE` to the specified account through the `addHeroGenerator` function.
- Admin can remove `HERO_GENERATOR_ROLE` from the specified account through the `removeHeroGenerator` function.
- Admin can add `BANNER_GENERATOR_ROLE` to the specified account through the `addBannerGenerator` function.
- Admin can remove `BANNER_GENERATOR_ROLE` from the specified account through the `removeBannerGenerator` function.
- Admin can add `ORB_GENERATOR_ROLE` to the specified account through the `addOrbGenerator` function.
- Admin can remove `ORB_GENERATOR_ROLE` for the specified account through the `removeOrbGenerator` function.
- Admin can add `ROLE` to the specified account through the `grantRole` function.
- Admin can remove `ROLE` for the specified account through the `revokeRole` function.

BlockLords/dynasty-smart-contract/NFTImportHub.sol

Deployers

- Deployers can initialize the contract Owner, `heroNft` , `bannerNft` , `orbNft` through the `constructor` function.

Owner

- Owner can set the `verifier` address through the `setVerifier` function.
- Owner can pause/unpause the contract through the `pause/unpause` function.

User

- User can transfer HeroNft to the dEaD address through the `importHeroNft` function and record the event for offline processing.
- User can transfer HeroNft to the dEaD address through the `importBannerNft` function and record the event for offline processing.

BlockLords/dynasty-smart-contract/Chest.sol

Deployers

- Deployers can initialize the contract Owner, `heroNft` , `bannerNft` , `orbNft` through the `constructor` function, and set the `initialOwner` to `DEFAULT_ADMIN_ROLE` .

Owner

- Owner can set a new `Season` through the `startSeason` function.
- Owner can set the `verifier` address through the `setVerifier` function.
- Owner can set `maxNFTsWithdrawal` through the `setMaxNFTsWithdrawal` function to limit the maximum limit for a single withdrawal.
- Owner can pause/unpause the contract through the `pause/unpause` function.

User

- Users can use the `openChest` function to open treasure chests in the game and obtain NFTs of the corresponding type from Mint.
- Users can use the `craftOrb` function to synthesize more advanced NFTs. They need to destroy NFTs of the specified type and mint new NFTs.
- User can use the `burnOrbForLRDS` function to destroy OrbNFT and trigger an event to record the number of `LRDS tokens` for offline processing.

Updated section:

- HeroNFT.sol

  Add factory's `mint` function for Mint HeroNFT.

  Add `setFactory` function to set `factory` address.

- BannerNFT.sol

  Add factory's `mint` function for Mint BannerNFT.

  Add `setFactory` function to set `factory` address.

- Dule.sol

  Added `seasonWithdraw` method for Claims rewards for a specific season.

4 Findings

CHE-1 Array Out-of-bounds

Severity: Major

Status: Fixed

Code Location:

contracts/game/Chest.sol#104

Descriptions:

The length of the array of `tokenIds` is not initialized, resulting in a return value that cannot be assigned and the function will fail to execute.

```
    uint256[] memory tokenIds;

    for (uint256 i = 0; i < nftTypeIndices.length; i++) {
        uint256 nftTypeIndex = nftTypeIndices[i];

        uint256 itemCode = itemCodes[i];

        tokenIds[i] = _mint(nftTypeIndex, itemCode);
    }
```

Suggestion:

It is recommended to set the array length and check it as follows, for example:

```
uint256[] memory tokenIds = new uint256[](nftTypeIndices.length);
```

Resolution:

Fix as suggested.

CHE-2 `burn` Function Cannot Be Called

Severity: Major

Status: Fixed

Code Location:

contracts/game/Chest.sol#235

Descriptions:

In the craftOrb/burnOrbForLRDS function, the contract only has authorized operation rights and does not have NFT ownership rights. The user has not transferred the NFT to the current contract. The burn function is called in the OrbNFT contract to determine the NFT owner. craftOrb cannot be called, and the function will fail to execute.

```
    nft.burn(nftId);
```

Suggestion:

It is recommended to modify it based on business logic.

Resolution:

Burn by transferring the NFT to the `dEaD` address.

```
nft.safeTransferFrom(msg.sender, 0x000000000000000000000000000000000000dEaD,
nftId);
```

CHE-3 Signature Nonce Is Not Increased

Severity: Minor

Status: Fixed

Code Location:

contracts/game/Chest.sol#102 239

Descriptions:

The `nonce[msg.sender]` in `message` is not updated after verification, which may cause signature reuse. The `nonce` mapping variable in OrbNFT will not be recorded in the chest contract, and calling the `mint` function through the factory will not verify the signature.

```
bytes32 message = keccak256(abi.encodePacked(msg.sender, _data, address(this),
nonce[msg.sender], _deadline, block.chainid));
```

Suggestion:

It is recommended to update nonce[msg.sender] after calling `verifySignature` in `openChest`.

Resolution:

Fix as suggested.

CHE-4 `pause` Is Not Implemented

Severity: Minor

Status: Fixed

Code Location:

contracts/game/Chest.sol#322;

contracts/game/Duel.sol#215;

contracts/game/Missions.sol#176;

contracts/game/NFTImportHub.sol#143

Descriptions:

The contract uses the `pause` function to change the `Pausable` state, but does not use the `whenNotPaused/whenPaused` modifier function.

```
function pause() public onlyOwner {
    Pausable._pause();
}

function unpause() public onlyOwner {
    Pausable._unpause();
}
```

Suggestion:

It is recommended to add modifications to the function where you want to use the `pause` function.

Resolution:

Fix as suggested.

CHE-5 `OrbNFT` Missing Permissions

Severity: Minor

Status: Fixed

Code Location:

contracts/game/Chest.sol#193

Descriptions:

The Chest contract cannot directly call the `OrbNFT.mint` function because the contract is not a `factory`.

```
uint256 mintedNftId = 0;
nonce[msg.sender]++;

mintedNftId = nft.mint(msg.sender, _quality);
```

The `factory` address at this time should be the `NFTFactory` contract.

```
function mint(address _to, uint256 _quality) external onlyFactory nonReentrant returns
(uint256)
```

Suggestion:

It is recommended to add the Chest contract mint channel or call `NFTfactory`.

Resolution:

Fix as suggested.

```
 mintedNftId = NftFactory(nftFactory).mintOrb(msg.sender, _quality);
```

DUE-1 NFT Missing Length Check

Severity: Minor

Status: Fixed

Code Location:

contracts/game/Duel.sol#166;

contracts/game/Chest.sol#105

Descriptions:

The contract in the project lacks a calldata length check, and NFT Mint will increase gas consumption very quickly as it cycles, which is not very rigorous in code.

```
    for (uint256 i = 0; i < nftTypeIndices.length; i++) {
      uint256 nftTypeIndex = nftTypeIndices[i];
      uint256 tokenId = tokenIds[i];
      _mint(nftTypeIndex, tokenId);
    }
```

Suggestion:

It is recommended to limit the number of single mints.

Resolution:

Increase the maximum limit and check the length.

```
require(nftTypeIndices.length <= maxNFTsWithdrawal, "Exceeds maximum allowed NFTs
per withdrawal");
```

NFA-1 Centralization Risk

Severity: Major

Status: Acknowledged

Code Location:

contracts/nfts/NftFactory.sol#148

Descriptions:

Centralization risk was identified in the smart contract.

- If Owner uses the `addAdmin` function to add multiple administrators, managers with `DEFAULT_ADMIN_ROLE` permissions can delete each other's permissions(`revokeRole` function).
- The quite valuable `orbNFT` and `factory` address can directly call the `mint` function to cast `orbNFT`. When the orbNFT contract Owner private key is leaked, it may cause certain risks.

Suggestion:

It is recommended that measures be taken to reduce the centralization issue.

Resolution:

Add event record.

NFT-1 Data Truncation

Severity: Major

Status: Fixed

Code Location:

contracts/game/NFTImportHub.sol#67,99

Descriptions:

Using a fixed array will not automatically truncate the calldata header, resulting in incorrect tokenID data.

```
(uint256[5] memory _nft) = abi.decode(_data, (uint256[5]));
```

POC:

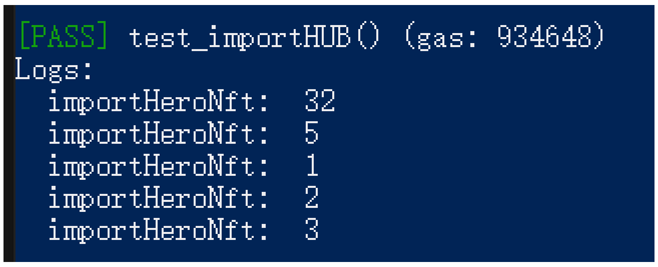
```
  bytes public data2;
  uint256[5] public _nft2;

  //function test() public returns (bytes memory _data) {
  function test() public returns (uint256[5] memory _nft) {
    uint256[] memory id = new uint256[](5);
    id[0] = 1;
    id[1] = 2;
    id[2] = 3;
    id[3] = 4;
    id[4] = 5;
    bytes memory _data = abi.encode(id);

    (uint256[5] memory _nft) = abi.decode(_data, (uint256[5]));

    data2 = _data;
    _nft2 = _nft;
  }
```

data

```
0x0000000000000000000000000000000000000000000000000000000000000002
0000000000000000000000000000000000000000000000000000000000000005
0000000000000000000000000000000000000000000000000000000000000001
```

```
0000000000000000000000000000000000000000000000000000000000000002
0000000000000000000000000000000000000000000000000000000000000003
0000000000000000000000000000000000000000000000000000000000000004
0000000000000000000000000000000000000000000000000000000000000005
```

Output results:



```
[PASS] test_importHUB() (gas: 934648)
Logs:
  importHeroNft:  32
  importHeroNft:  5
  importHeroNft:  1
  importHeroNft:  2
  importHeroNft:  3
```

Suggestion:

It is recommended to modify it to a dynamic array and add a length check.

For example

```
require(_nft.length == 5,"not the correct length");
(uint256[] memory _nft) = abi.decode(_data, (uint256[]));
```

Resolution:

Fix as suggested.

NFT-2 Calldata Missing Check

Severity: Medium

Status: Fixed

Code Location:

contracts/game/NFTImportHub.sol#62

Descriptions:

The `importHeroNft/importBannerNft` function does not check the calldata data consistency, resulting in function execution errors or loop errors.

```
  function importHeroNft(bytes calldata _data, uint256 _deadline, uint8 _v, bytes32 _r,
bytes32 _s) external nonReentrant whenNotPaused {
      // Ensure signature has not expired
      require(_deadline >= block.timestamp, "Signature has expired");

      // Decode the data containing NFT IDs
      (uint256[5] memory _nft) = abi.decode(_data, (uint256[5]));

...

      for(uint i = 0; i < 5; i++){
         if(_nft[i] > 0){
...
         }
      }
...
  }
```

Suggestion:

It is recommended to add data length check.

Resolution:

Fix as suggested.

```
 require(_nft.length == 5, "Incorrect NFT length");
```

ONF-1 Incorrect Error Code

Severity: Minor

Status: Fixed

Code Location:

contracts/nfts/OrbNFT.sol#93

Descriptions:

The error alert is incorrect.

```
require(recover == verifier, "Verification failed about mint hero nft");
```

Suggestion:

It is recommended to change to the correct orbNFT prompt.

Resolution:

Fix as suggested.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.