# RentFun
# Audit Report

Mon Dec 04 2023

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

ScaleBit

# RentFun Audit Report

## 1 Executive Summary

### 1.1 Project Information

| | |
|---|---|
| Description | First NFT Rental Protocol for arbitrum Games. |
| Type | Lending |
| Auditors | ScaleBit |
| Timeline | Wed Nov 15 2023 - Mon Dec 04 2023 |
| Languages | Solidity |
| Platform | Arbitrum |
| Methods | Architecture Review, Unit Testing, Manual Review |
| Source Code | https://github.com/RentFun/summer |
| Commits | ba4ff43a7827562ac9d82da7384fcd87596550f4<br>21f24c9ea1e3718c0823cc23171952d9e3a634d7<br>47c0ad141870fd55504ac8e3c7f31125f01814bd |

## 1.2 Files in Scope

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

| ID | File | SHA-1 Hash |
|---|---|---|
| RFU | contracts/RentFun/RentFun.sol | ebe76595391f77afee48c4bd64eae183300fc775 |
| IRF | contracts/RentFun/interfaces/IRentFun.sol | b9cfbf4e3722be9e413b5a6ac63a61bf35cee681 |
| IVA | contracts/RentFun/interfaces/IVault.sol | 1ccf3c31de5b5a41350198346bedcd851b707455 |
| IVM | contracts/RentFun/interfaces/IVaultManager.sol | e5fef1de3449a244e68ad54373754974c367ba71 |
| RFH | contracts/RentFun/RentFunHelper.sol | d5b1c6720c7ea0b1e350e5304bc73b37b580a5fb |
| VMA | contracts/RentFun/VaultManager.sol | 011d9a0a4b45f69f610972567dd461545b88ef64 |
| VAU | contracts/RentFun/Vault.sol | 410cae0a799b84afe2a43a4197d79da897e26ec6 |
| NFT | contracts/Token/NFToken.sol | 0777951e9a406c3b71812fcef674fa06448a7601 |
| WBI | contracts/Token/WonderBird.sol | af73a42aac23a38ac2f966d7848c5439266b7c76 |
| RTO | contracts/Token/RentToken.sol | a1707643567a98eca7a85dcd3f6ab739e7eec294 |

## 1.3 Issue Statistic

| Item | Count | Fixed | Acknowledged |
| --- | --- | --- | --- |
| Total | 9 | 4 | 5 |
| Informational | 0 | 0 | 0 |
| Minor | 2 | 2 | 0 |
| Medium | 4 | 0 | 4 |
| Major | 3 | 2 | 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 RentFun to identify any potential issues and vulnerabilities in the source code of the RentFun 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 9 issues of varying severity, listed below.

| ID | Title | Severity | Status |
| --- | --- | --- | --- |
| RFH-1 | Use `abi.encode` instead of `abi.encodePacked` | Minor | Fixed |
| RFU-1 | `rent` Function Can Be Front-run | Major | Fixed |
| RFU-2 | `lend` Function Design Issues | Major | Acknowledged |
| RFU-3 | Token Can Be Rented Multiple Times | Major | Fixed |
| RFU-4 | Missing Check On `collection` address | Medium | Acknowledged |
| RFU-5 | `RentFun` Contract May Have `ptnFee` Remaining | Medium | Acknowledged |
| RFU-6 | `Initialize` Could Be Front-Run | Medium | Acknowledged |
| VMA-1 | Initialize Issue | Medium | Acknowledged |
| VMA-2 | Missing Events For Important Parameter Updates | Minor | Fixed |

# 3 Participant Process

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

**Admin**

- The admin can set the max vault of user can create by calling the `setMaxVaultNum()` function.
- The admin can set the `Rentfun` address `setRentfun()` function.
- The admin has the authority to call the `setCommission` and `setVipCommission` functions to set `commission_` and `vipCommission_`.
- The admin has the ability to invoke the `setPartner` function to set `Partner`.
- The admin has the capability to invoke the `setTreasure` function to change `setTreasure` addresses.
- The admin has the capability to invoke the `setVaultManager` function to change `vaultManager` addresses.
- The admin has the capability to invoke the `setWonderBird` function to change `wonderBird` addresses.

**User**

- Users can invoke the `create` function to create a new vault.
- Users can call `removeVault` function to remove the vault of user.
- Users have the option to call the `transferERC721` function to transfer NFT to vault.
- Users can use the `transferERC20` function to transfer ERC20 token to vault.
- Users can invoke the `transferETH` function to transfer ETH to vault.
- Users can use the `lend` function to lend a token.
- Users can use the `rent` function to rent a token.
- Lender can call `claimRentFee` to get the fee of `lend bid`.

# 4 Findings

## RFH-1 Use `abi.encode` instead of `abi.encodePacked`

Severity: Minor

Status: Fixed

Code Location:

contracts/RentFun/RentFunHelper.sol#133-144

Descriptions:

Use `abi.encode()` instead which will pad items to 32 bytes, which will prevent hash collisions (e.g. `abi.encodePacked(0x123,0x456)` => `0x123456` => `abi.encodePacked(0x1,0x23456)`, but `abi.encode(0x123,0x456)` => `0x0...1230...456`). Unless there is a compelling reason, `abi.encode` should be preferred.

Suggestion:

It is recommended to use `abi.encode` as preferred.

Resolution:

The client followed the suggestion and fixed this issue.

## RFU-1 `rent` Function Can Be Front-run

Severity: Major

Status: Fixed

Code Location:

contracts/RentFun/RentFun.sol#71

Descriptions:

The `rent()` function has a known race condition that can lead to token theft. If a renter calls the `rent` function a second time, the lender can front-run the transaction and call `lend()` again to modify the lend bid and raise the bid fee, while the transaction of receive will still be packed, so this will result in the loss of the renter's token.

Suggestion:

It is recommended to add a parameter to limit the maximum number of tokens a renter can pay. This will help prevent users from losing funds from front-running attacks.

Resolution:

The client followed the suggestion and fixed this issue.

## RFU-2 `lend` Function Design Issues

Severity: Major

Status: Acknowledged

Code Location:

contracts/RentFun/RentFun.sol#70

Descriptions:

The `lend` function allows the owner of the `tokenID` to rent out his nft and set the rental fee, then the bid information and in basic information of the token will be recorded in the `lendTokens` and `lendBids`. However, `lend` does not judge whether the token is already in the state of renting when lender call `lend`, which means that the owner can call the lend function again to modify the information of lend, such as modifying the `maxEndTime` of the token (in addition to the `tokenHash`).

Suggestion:

It is recommended to determine the state of the corresponding `tokenHash` before lend.

Resolution:

According to the protocol design `lend` can be called multiple times by the owner of the token to modify the relevant bid information.

# RFU-3 Token Can Be Rented Multiple Times

Severity: Major

Status: Fixed

Code Location:

contracts/RentFun/RentFun.sol#99

Descriptions:

After the lender calls the `lend` function, other users can call the `rent` and pay the corresponding fee to rent the token, but the `rent` function does not check the state of the token corresponding to the current `tokenHash` and whether it has been rented out or not, so it will result that a token can be rented out more than once.

Suggestion:

It is recommended to add a state check of the `tokenHash` corresponding to the token in the rent function.

Resolution:

The client followed the suggestion and fixed this issue.

# RFU-4 Missing Check On `collection` address

Severity: Medium

Status: Acknowledged

Code Location:

contracts/RentFun/RentFun.sol#71

Descriptions:

Due to the lack of checking of the `collection` address, the lender can pass in any `collection` address when calling the `lend` function, and it is possible that the collection is an invalid address or a dangerous contract address, which may result in an unknown risk.

Suggestion:

It is recommended to take relevant methods to limit the `collection` address to fix the issue.

Resolution:

The client replied that there are no restrictions on `collections` at this time, and restrictions on `collection` addresses may be added in the future.

# RFU-5 `RentFun` Contract May Have `ptnFee` Remaining

Severity: Medium

Status: Acknowledged

Code Location:

contracts/RentFun/RentFun.sol#158

Descriptions:

When the `claimRentFee` function is called by the lender, it will distribute part of the rental fee to the protocol, and part of this fee will be given to the partners. But if the receiver `address` of the partners is 0, and the `share` is not set to 0, the `claimRentFee` function will not do anything with this part of this fee, and the token will be locked in the `RentFun` contract.

Suggestion:

It is recommended to make sure that when adding a partner address, when `receiver` is set to 0, `share` is 0 as well.

Resolution:

Setting `initialized` to true prevents the user from invoking the `rentfun` contract directly, but instead interacts with it through a proxy contract.

## RFU-6 `Initialize` Could Be Front-Run

Severity: Medium

Status: Acknowledged

Code Location:

contracts/RentFun/RentFun.sol#64;

contracts/RentFun/VaultManager.sol#28

Descriptions:

In the contract, by calling the `initialize` function to initialize the contracts, there is a potential issue that malicious attackers preemptively call the initialize function to initialize and there is no access control verification for the initialize functions.

Suggestion:

It is suggested that the `initialize` function can be called in the same transaction immediately after the contract is created to avoid being maliciously called by the attacker.

Resolution:

The client replied that the deployed contract and `initialize` will be called in one transaction.

# VMA-1 Initialize Issue

Severity: Medium

Status: Acknowledged

Code Location:

contracts/RentFun/VaultManager.sol#25;

contracts/RentFun/RentFunHelper.sol#40;

contracts/RentFun/RentFun.sol#61

Descriptions:

`VaultManager`, `RentFunHelper` and `RentFun` contracts are deployed with the constructor setting the value of `initialized` to true, which can cause the admin to fail when initializing the contract to set global variables.

Suggestion:

It is recommended to modify the constructor so that the contract can be initialized correctly.

# VMA-2 Missing Events For Important Parameter Updates

Severity: Minor

Status: Fixed

Code Location:

contracts/RentFun/VaultManager.sol#56-60

Descriptions:

We found that when important parameters are updated in the project, the function doesn't emit the update event, so we suggest emitting the event in time, so to notify the user or chain off programs.

Suggestion:

It is recommended to emit the corresponding event in time when updating the important parameter.

Resolution:

The client followed the suggestion and fixed this issue.

# 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.