Foundation Exhibition, Scheduling & Early Access Solo Audit by Lambda
Findings & Analysis Report

2022-12-14

Table of contents

Overview

About C4

Code4rena (C4) is an open organization consisting of security researchers, auditors, developers, and individuals with domain expertise in smart contracts.

A C4 Solo Audit is an event where a top Code4rena contributor, commonly referred to as a warden or a team, reviews, audits and analyzes smart contract logic in exchange for a bounty provided by sponsoring projects.

During the Solo Audit outlined in this document, C4 conducted an analysis of the Foundation code. The audit took place between November 21-28, 2022.

Wardens

Audit completed by Lambda.

Final report assembled by itsmetechjay.

Summary

The Foundation Exhibition, Scheduling & Early Access Solo Audit yielded 3 MEDIUM vulnerabilities. There were also 7 informational findings and 1 gas optimization reported.

The codebase in question had already undergone two prior Code4rena contests. Since the last audit, exhibitions (listing NFTs with a curator that gets part of the final sale price) and the possibility to set early access periods for drop markets were added.

Scope

Code reviewed consisted of the following contracts:

Contract Name	SLOC	Purpose
NFTDropMarket	63	The main / top-level contract for all drop market tools on Foundation.
FoundationTreasuryNode	34	A wrapper for communicating with the treasury contract which collects Foundation fees and defines the central roles.
FETHNode	35	A wrapper for communicating with the FETH contract.
MarketSharedCore	7	A base class for Foundation market contracts to define functions that other market contract may implement or extend.
NFTDropMarketCore	4	A base class for the drop specific market contract to define functions that other mixins may implement or extend.
SendValueWithFallbackWithdraw	19	A helper function to handle funds transfers.
MarketFees	331	Distributes revenue from sales.
Gap500	4	A placeholder contract leaving room for new mixins to be added to the future in an upgrade safe fashion.
Gap10000	4	A placeholder contract leaving room for new mixins to be added to the future in an upgrade safe fashion.
NFTDropMarketFixedPriceSale	253	Allows creators to list a drop collection for sale at a fixed price point.

Contract Name	SLOC	Purpose
ArrayLibrary	17	Helper functions for resizing arrays.
Constants	10	Shared constant values used by various mixins.
MerkleAddressLibrary	9	A wrapper for validating merkle proofs for a tree containing a list of addresses.
TimeLibrary	9	Simple time checks to improve code readability & ensure consistency.
ExhibitionMarketMock	63	A subset of the NFTMarket contract including auctions & exhibitions.
NFTMarketCore	44	A base class for the market contract to define functions that other mixins may implement or extend.
NFTMarketExhibition	127	Adds exhibitions to the marketplace.
NFTMarketAuction	14	A base for reserve auctions which may be shared with other auction types in the future.
NFTMarketReserveAuction	319	Adds support for reserve auctions to the marketplace.
Total 19 files	1,366	

Severity Criteria

C4 assesses the severity of disclosed vulnerabilities according to a methodology based on **OWASP standards**.

Vulnerabilities are divided into three primary risk categories: high, medium, and low/non-critical.

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious Input Handling
- Escalation of privileges
- Arithmetic
- Gas use

Further information regarding the severity criteria referenced throughout the submission review process, please refer to the documentation provided on [the C4 website](#).

Medium Risk Findings (3)

[M-01] `MarketFees` : Seller referrer fee not paid when no creator royalty recipients exist for a sale

The logic to calculate the seller referrer fee is within the following `if` block:

```
if (creatorRecipients.length != 0 || assumePrimarySale) {
  ...
}
```

When `creatorRecipients.length == 0` and `assumePrimarySale == false` (which is the case for `ExhibitionMarketMock`), the returned `sellerReferrerFee` will always be zero, no matter which value is passed for `sellerReferrerTakeRateInBasisPoints` . Therefore, the exhibition curator will not get the configured fee in such a scenario, although the seller referral fee should not depend on the existence of creator royalty recipients.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/shared/MarketFees.sol#L530

Recommended Mitigation Steps

Calculate the seller referrer fee also when no creator royalty recipients are defined (like it is done for the buyer referrer fee).

[M-02] `MarketFees` : Primary sales not detected in some scenarios

When the seller is also a recipient of the creator royalties, a primary sale is assumed and `sellerRev` is set to 0:

```
if (creatorRecipients[i] == seller) {
  // If the seller is any of the recipients defined,
  creatorRev += sellerRev;
  sellerRev = 0;
}
```

However, this loop is exited prematurely when one of the `creatorShares` entries is greater than `BASIS_POINTS` :

```
if (creatorShares[i] > BASIS_POINTS) {
  // If the numbers are >100% we ignore the fee reci
  totalShares = 0;
  break;
}
```

Therefore, there can be scenarios where it is not detected that the seller is a recipient of the creator royalties and a secondary sale is assumed. For instance, if `creatorRecipients` is `[address(Bob), address(Alice), seller]` and `creatorShares` is `[100, 10_001, 100]` , the loop will `break` in the second iteration and `sellerRev` will not be set to 0.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/shared/MarketFees.sol#L555

Recommended Mitigation Steps

Continue looping to detect if the seller is a creator royalty recipient, even if one `creatorShares` entry was invalid.

[M-03] `MarketFees` : Seller referrer fee can underflow

In `MarketFees._getFees`, the `sellerReferrerFee` is deducted from `creatorRev` or `sellerRev` (depending on if the sale is primary / secondary):

```
if (sellerReferrerTakeRateInBasisPoints != 0) {
  sellerReferrerFee = (price * sellerReferrerTakeRateInBas

  // Subtract the seller referrer fee from the seller reve
  if (sellerRev == 0) {
    // If the seller revenue is 0, this is a primary sale
    creatorRev -= sellerReferrerFee;
  } else {
    sellerRev -= sellerReferrerFee;
  }
}
```

`sellerReferrerTakeRateInBasisPoints` can be up to 5,000 (50%), because the exhibition take rate (which is the only case when a non-zero value is passed) needs to be smaller than `MAX_TAKE_RATE`. On the other hand, it is only enforced that the protocol fee plus the creator royalty percentage is smaller than 100%:

```
if (
  protocolFeeInBasisPoints < BASIS_POINTS / BUY_REFERRER_FEE
  protocolFeeInBasisPoints + BASIS_POINTS / CREATOR_ROYALTY_
) {
  /* If the protocol fee is invalid, revert:
   * Protocol fee must be greater than the buy referrer fee
   * The protocol fee must leave room for the creator royalt
   */
  revert NFTMarketFees_Invalid_Protocol_Fee();
}
```

Because of that, there are valid settings where the calculation will underflow, leading to sales that do not succeed. For instance, we can have a protocol fee of 20% and creator royalties of 35%, in which case `sellerRev` would be 45% of the overall price. If the exhibition take rate is then >45%, the calculation will underflow.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/shared/MarketFees.sol#L534

Recommended Mitigation Steps

There are two options to solve the problem:

- Enforce that the protocol fee plus the creator royalty percentage is less than 50%.
- Handle the underflow case explicitly and for instance cap the seller referrer fee if it would exceed the seller revenue.

Informational Findings (7)

[Info-01] `NFTMarketExhibition`: `_getAndRemoveNftFromExhibition` does not emit `NftRemovedFromExhibition`

Unlike the function `_removeNftFromExhibition`, `_getAndRemoveNftFromExhibition` does not emit the event `NftRemovedFromExhibition` when an NFT is removed from an exhibition. This can be problematic for blockchain indexing solutions that rely on events (for instance, to track which NFTs are listed in which exhibition), as they will not detect the removal when an NFT is sold via a reserve auction.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/nftMarket/NFTMarketExhibition.sol#L199

Recommended Mitigation Steps

Also emit the event in `_getAndRemoveNftFromExhibition`

[Info-02] `NFTMarketExhibition` : Behavior of `getExhibition` for non-existing exhibition IDs

When a non-existing exhibition ID is passed to `getExhibition`, the default values (empty string, `address(0)`, and 0) will be returned for all parameters. Because of that, it may not be clear for integrators how to check the existence of an exhibition. As an empty string is a valid value for the name and the take rate can be 0, the only reliable way is to check if the curator is equal to `address(0)`. If an integrator decides to use another value, the check will be unreliable.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/nftMarket/NFTMarketExhibition.sol#L224

Recommended Mitigation Steps

Either revert for non-existing exhibitions or document clearly how a non-existing exhibition will be returned.

[Info-03] `NFTMarketReserveAuction` : Event `ReserveAuctionCreated` does not include exhibition ID

The event `ReserveAuctionCreated` that is emitted within `createReserveAuctionV2` does not include the exhibition ID. Because the exhibition ID determines the seller referrer fee, this information may be important for blockchain indexing solutions or other applications that consume these events.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/nftMarket/NFTMarketReserveAuction.sol#L309

Recommended Mitigation Steps

Consider adding the exhibition ID to the event.

[Info-04] `NFTDropMarketFixedPriceSale` : Provided start times can be arbitrarily far in the future

While `_createFixedPriceSale` validates that the provided start times (early access and general availability) are not in the past, they can be arbitrarily far (up to February 2106) in the future. Therefore, when a user submits a completely wrong timestamp by mistake, the sale will still be created.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/nftDropMarket/NFTDropMarketFixedPriceSale.sol#L295

Recommended Mitigation Steps

To catch user errors, consider adding an upper limit on the time delta (for instance 4 years).

[Info-05] `NFTDropMarketFixedPriceSale` : `getFixedPriceSale` may return 0 for `generalAvailabilityStartTime` of valid sale

As an optimization, `_createFixedPriceSale` does not store the `generalAvailabilityStartTime` when it is equal to the current block timestamp (because the business logic does not need the value then):

```
        if (generalAvailabilityStartTime != block.timestamp) {
          // If starting now we don't need to write to storage
          saleConfig.generalAvailabilityStartTime = generalAvailabil
        }
```

However, this also means that `getFixedPriceSale` will set `generalAvailabilityStartTime` to 0 for such sales. Therefore, there is for instance no way to figure out the time that has passed since the general availability start time (which applications that build on top of Foundation might want to do) in these scenarios.

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/nftDropMarket/NFTDropMarketFixedPriceSale.sol#L361

Recommended Mitigation Steps

Consider still setting `generalAvailabilityStartTime` in these cases, even if gas usage increases slightly.

[Info-06] `NFTDropMarketFixedPriceSale` : Fixed merkle root per sale may be restricting

The creator of a sale cannot add a new merkle root for a sale or change the current one. This may be very restricting in some scenarios. For instance, it could happen that an error occurred during the calculation of the root or that one address needs to be changed (for instance, because a user lost his private key).

Link To Affected Code

https://github.com/f8n/fnd-contracts-staging/blob/c5ca7eb50e3fe3221bda49168cae8b4bbd98f4ac/contracts/mixins/nftDropMarket/NFTDropMarketFixedPriceSale.sol#L367

Recommended Mitigation Steps

According to a comment in `NFTDropMarketFixedPriceSale` , multiple merkle roots per sale are planned as a future feature. However, as a temporary solution, the possibility to at least replace the current one (e.g., where the seller has to provide the old one and the new one) could help in the described situations.

[Info-07] Undocumented parameters

In a two places, a parameter is missing in the documentation:

- NFTMarketExhibition: `exhibitionId` within the event `NftRemovedFromExhibition`

- NFTMarketReserveAuction: The parameter `referrer` of `placeBidV2` is not documented.

Gas Optimizations (1)

[G-01] `NFTMarketExhibition.createExhibition`: Unnecessary storage read

In line 138 of `NFTMarketExhibition`, instead of reading `idToExhibition[exhibitionId].curator`, `curator` can be set to `msg.sender`, resulting in the following reduction of gas usage when creating exhibitions:

```
7,10c7,10
<     108'739 - w/ 1
<     132'430 - w/ 2
<     203'497 - w/ 5
<     1'269'704 - w/ 50
---
>     108'924 - w/ 1
>     132'615 - w/ 2
>     203'682 - w/ 5
>     1'269'889 - w/ 50
50c50
<     125'699 - createFixedPriceSaleWithEarlyAccessAllowlist
---
>     125'711 - createFixedPriceSaleWithEarlyAccessAllowlist
```

Disclosures

C4 is an open organization governed by participants in the community.

C4 does not provide any guarantee or warranty regarding the security of this project. All smart contract software should be used at the sole risk and responsibility of users.

Top