SECURITY AUDIT REPORT

for project: **Foundation**

by auditor: **Odd Sequence**

issued: **29 December 2022**

ONE-PAGER

ABOUT

PROJECT

Name	Foundation Drop
Description	Foundation Drop contracts aim to build NFT collection, dropping, and selling.

AUDITOR

Team	Odd Sequence
Specialists	Two specialists

TASK

Scope links	Github page: - page [link] - commit b4e1dc6a9a434d576cb8f74676e7dbb65dacc98d [link]
Scope Description	Smart-contracts Transaction flow Connections with external contracts
Not-in-Scope	FETH.sol PercentSplitETH.sol FoundationTreasury.sol mocks/* Best-Practice code remarks Standardized libraries Gas efficiency (if not dangerous)
Networks	EVM networks
Languages	Solidity
Deployment env.	Hardhat framework project
Timeline	bughunting: 11.08.2022 - 15.08.2022
To Do	To find some vulnerabilities of the code in the Scope, that will be possible to find given the time and efforts of specialists
Done	The auditor has found some vulnerabilities. They are described in the report as “Findings”. The report does not cover all vulnerabilities possible.
Terms	Auditors received a reward from Code4rena after the Rigor Finance contest on Code4rena. Auditors reported findings to Code4rena and received acceptance for some of them.

FINDINGS

2 findings

MEDIUM **acknowledged**	NFTDropMarket.sol accept any NFT contracts for sales, take money from buyers, but don't check NFTs were minted
MEDIUM **acknowledged**	Input [limitperAccount] in NFTDropMarket.createFixedPriceSale() is easily avoidable in one transaction

1/2
MEDIUM

@
NFTDropMarket.sol

Acknowledged

`NFTDropMarket.sol` accepts any NFT contracts for sales, takes money from buyers, but doesn't check NFTs were minted

Description

In the https://os.foundation.app/docs/creator-tools/drop we can outline that:

1. Any NFT can be added to the sale, not only deployed by the NFTCollectionFactory (NFTDropMarket.createFixedPriceSale() is correct here, it accept any NFT, at least those implementing INFTDropCollectionMint interface)
2. The market itself is "A collection primitive for delegated minting" So it is the market's role to arrange minting for payments.

`NFTDropMarket` perfectly takes payments from buyers, but does not check if the mint is successful after the payment. Mints will be ok if NFT contracts are deployed through `NFTCollectionFactory.sol`, but for malicious NFT it is not a guarantee.

Check for the interface implemented is not enough.

@ NFTDropMarketFixedPriceSale.sol, lines 118-157 and lines 170-219

```
  function createFixedPriceSale(
    address nftContract,
    uint80 price,
    uint16 limitPerAccount
  ) external {

    // Confirm the drop collection is supported
    if (!nftContract.supportsInterface(type(INFTDropCollectionMint).interfaceId)) {
      revert NFTDropMarketFixedPriceSale_Must_Support_Collection_Mint_Interface();
    }
    if (INFTDropCollectionMint(nftContract).numberOfTokensAvailableToMint() == 0) {
      revert NFTDropMarketFixedPriceSale_Must_Not_Be_Sold_Out();
    }

    // Use the AccessControl interface to confirm the msg.sender has permissions to list.
    if (!IAccessControl(nftContract).hasRole(DEFAULT_ADMIN_ROLE, msg.sender)) {
      revert NFTDropMarketFixedPriceSale_Only_Callable_By_Collection_Owner();
    }
    // And that this contract has permission to mint.
    if (!IAccessControl(nftContract).hasRole(MINTER_ROLE, address(this))) {
      revert NFTDropMarketFixedPriceSale_Mint_Permission_Required();
    }

    // Validate input params.
    if (limitPerAccount == 0) {
      revert NFTDropMarketFixedPriceSale_Limit_Per_Account_Must_Be_Set();
    }
    // Any price is supported, including 0.

    // Confirm this collection has not already been listed.
    FixedPriceSaleConfig storage saleConfig = nftContractToFixedPriceSaleConfig[nftContract];
    if (saleConfig.seller != payable(0)) {
      revert NFTDropMarketFixedPriceSale_Must_Not_Have_Pending_Sale();
    }

    // Save the sale details.
    saleConfig.seller = payable(msg.sender);
    saleConfig.price = price;
    saleConfig.limitPerAccount = limitPerAccount;
    emit CreateFixedPriceSale(nftContract, saleConfig.seller, saleConfig.price,
saleConfig.limitPerAccount);
  }
...

  function mintFromFixedPriceSale(
    address nftContract,
    uint16 count,
    address payable buyReferrer
  ) external payable returns (uint256 firstTokenId) {

    // Validate input params.
```

```
    if (count == 0) {
      revert NFTDropMarketFixedPriceSale_Must_Buy_At_Least_One_Token();
    }

    FixedPriceSaleConfig memory saleConfig = nftContractToFixedPriceSaleConfig[nftContract];

    // Confirm that the buyer will not exceed the limit specified after minting.
    if (IERC721(nftContract).balanceOf(msg.sender) + count > saleConfig.limitPerAccount) {
      if (saleConfig.limitPerAccount == 0) {
        // Provide a more targeted error if the collection has not been listed.
        revert NFTDropMarketFixedPriceSale_Must_Have_Sale_In_Progress();
      }
      revert NFTDropMarketFixedPriceSale_Cannot_Buy_More_Than_Limit(saleConfig.limitPerAccount);
    }

    // Calculate the total cost, considering the `count` requested.
    uint256 mintCost;
    unchecked {
      // Can not overflow as 2^80 * 2^16 == 2^96 max which fits in 256 bits.
      mintCost = uint256(saleConfig.price) * count;
    }

    // The sale price is immutable so the buyer is aware of how much they will be paying when their tx
is broadcasted.
    if (msg.value > mintCost) {
      // Since price is known ahead of time, if too much ETH is sent then something went wrong.
      revert NFTDropMarketFixedPriceSale_Too_Much_Value_Provided(mintCost);
    }
    // Withdraw from the user's available FETH balance if insufficient msg.value was included.
    _tryUseFETHBalance(mintCost, false);

    // Mint the NFTs.
    firstTokenId = INFTDropCollectionMint(nftContract).mintCountTo(count, msg.sender);

    // Distribute revenue from this sale.
    (uint256 totalFees, uint256 creatorRev, ) = _distributeFunds(
      nftContract,
      firstTokenId,
      saleConfig.seller,
      mintCost,
      buyReferrer
    );

    emit MintFromFixedPriceDrop(nftContract, msg.sender, firstTokenId, count, totalFees, creatorRev);
  }
```

Lines

[link], [link]

Exploit Scenario

`NFTDropMarket` function only checks that the NFT implements INFTDropCollectionMint interface.

But for malicious NFT contracts, there are still too many options to write something bad - like mining 0 NFTs, after taking payments on NFTDropMarket.

So the steps are:

- Add malicious NFT contract to sale, through NFTDropMarket.createFixedPriceSale()
- It is listed
- When buyers trigger NFTDropMarket.mintFromFixedPriceSale(), ETH is taken perfectly from buyers, then distributed, but a malicious NFT contract can react as it desires, like not minting NFTs.

Recommendation

In `NFTDropMarket.mintFromFixedPriceSale()` check that the mint happened (like balanceOf check, or anything else). Or consider additional checks when adding NFTs to sales in `NFTDropMarket.createFixedPriceSale()`

2/2
MEDIUM

@
NFTDropMarket.sol

Acknowledged

Input [`limitperAccount`] in `NFTDropMarket.createFixedPriceSale()` is easily avoidable in one transaction

Description

It is designed to cap one buy per account by `limitperAccount` configured when sale is created.

It should revert with `NFTDropMarketFixedPriceSale_Cannot_Buy_More_Than_Limit()`
if `(IERC721(nftContract).balanceOf(msg.sender) + count > saleConfig.limitPerAccount)`

Thus to pass this check IERC721(nftContract).balanceOf(msg.sender) should be minimized. It is very easy to do in one transaction, if a buyer is a smart-contract (buy->transfer->buy->transfer)

As a result, anyone can buy as much NFT as he wishes, even all the collection. But the seller expects that the limit works.

@ NFTDropMarketFixedPriceSale.sol, lines 170-219

```
  function mintFromFixedPriceSale(
   address nftContract,
   uint16 count,
   address payable buyReferrer
 ) external payable returns (uint256 firstTokenId) {

   // Validate input params.
   if (count == 0) {
     revert NFTDropMarketFixedPriceSale_Must_Buy_At_Least_One_Token();
   }

   FixedPriceSaleConfig memory saleConfig = nftContractToFixedPriceSaleConfig[nftContract];

   // Confirm that the buyer will not exceed the limit specified after minting.
   if (IERC721(nftContract).balanceOf(msg.sender) + count > saleConfig.limitPerAccount) {
     if (saleConfig.limitPerAccount == 0) {
       // Provide a more targeted error if the collection has not been listed.
       revert NFTDropMarketFixedPriceSale_Must_Have_Sale_In_Progress();
     }
     revert NFTDropMarketFixedPriceSale_Cannot_Buy_More_Than_Limit(saleConfig.limitPerAccount);
   }

   // Calculate the total cost, considering the `count` requested.
   uint256 mintCost;
   unchecked {
     // Can not overflow as 2^80 * 2^16 == 2^96 max which fits in 256 bits.
     mintCost = uint256(saleConfig.price) * count;
   }

    // The sale price is immutable so the buyer is aware of how much they will be paying when their tx
is broadcasted.
   if (msg.value > mintCost) {
     // Since price is known ahead of time, if too much ETH is sent then something went wrong.
     revert NFTDropMarketFixedPriceSale_Too_Much_Value_Provided(mintCost);
   }
   // Withdraw from the user's available FETH balance if insufficient msg.value was included.
   _tryUseFETHBalance(mintCost, false);

   // Mint the NFTs.
   firstTokenId = INFTDropCollectionMint(nftContract).mintCountTo(count, msg.sender);

   // Distribute revenue from this sale.
   (uint256 totalFees, uint256 creatorRev, ) = _distributeFunds(
     nftContract,
     firstTokenId,
     saleConfig.seller,
     mintCost,
     buyReferrer
   );

   emit MintFromFixedPriceDrop(nftContract, msg.sender, firstTokenId, count, totalFees, creatorRev);
 }
```

Lines

[link]

Exploit Scenario

Steps - in one transaction from a smart-contract:

1. Buy one
2. Transfer this one somewhere
3. Buy new one
4. Transfer again

...and so one

n. Transfer all to the one EOA

Result - as many tokens per one account as desired.

Recommendation

Options:

- delete this input, accept that it is not possible to limit buys per user
- only EOA should buy
- block sales if tx.origin does not change
- block transfers when sale is on
- something more smart, it is many options possible

DISCLOSURES

CODE IN THE SCOPE

GITHUB

Link	https://github.com/code-423n4/2022-08-foundation/tree/b4e1dc6a9a434d576cb8f74676e7dbb65dacc98d
Contracts	NFTCollectionFactory.sol NFTCollection.sol NFTDropCollection.sol NFTDropMarket.sol

REPORT CHANGELOG

The first version of findings can be checked via this links:

code-423n4/2022-08-foundation-findings/issues/237
code-423n4/2022-08-foundation-findings/issues/263

The later versions are published on Odd Sequence Github page, where changes can be tracked:
https://github.com/oddsequence

FINDINGS STATUS

MEDIUM

1/1	Acknowledged
2/2	Acknowledged

DISCLOSURES

GENERAL DISCLOSURES

Odd Sequence typically receives compensation from one or more clients (the “Clients”) for performing the analysis contained in these reports (the “Reports”). The Reports may be distributed through other means, including via Github, website mainpage and other distributions.

The Reports are not an endorsement or indictment of any particular project or team, and the Reports do not guarantee the security of any particular project. This Report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any Third-Party in any respect, including regarding the bugfree nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. No third party should rely on the Reports in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. Specifically, for the avoidance of doubt, this Report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. Odd Sequence owes no duty to any Third-Party by virtue of publishing these Reports.

PURPOSE OF REPORTS

The Reports and the analysis described therein are created solely for Clients and published with their consent. This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The scope of our review is limited to a review of code and only the code we note as being within the scope of our review within this report. Any code itself presents unique and unquantifiable risks as the programming languages itself remain under development and is subject to unknown risks and flaws. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. The review does not extend to the compiler layer, or any other areas beyond specified code that could present security risks. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty.

Odd Sequence makes the Reports available to parties other than the Clients (i.e., “third parties”) – on its website. Odd Sequence hopes that by making these analyses publicly available, it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.

LINKS TO OTHER WEB SITES

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Odd Sequence. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Odd Sequence is not responsible for the content or operation of such Web sites, and that Odd Sequence shall have no liability to you or any other person or entity for the use of third party Web sites. Except as described below, a hyperlink from this web Site to another web site does not imply or mean that Odd Sequence endorses the content on that Web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the Reports. Odd Sequence assumes no responsibility for the use of third party software on the Web Site and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

TIMELINESS OF CONTENT

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Odd Sequence; however, Odd Sequence does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

NOTICE OF CONFIDENTIALITY

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Odd Sequence. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Odd Sequence. Odd Sequence assumes no responsibility for the use of third-party software on the website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software. In some instances, Odd Sequence may perform penetration testing or infrastructure assessments depending on the scope of the particular engagement.

No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.