METATRUST

Draft
Security Assessment for

69-2021-12-nftx (1OK-ANC) (1OK-FLB)

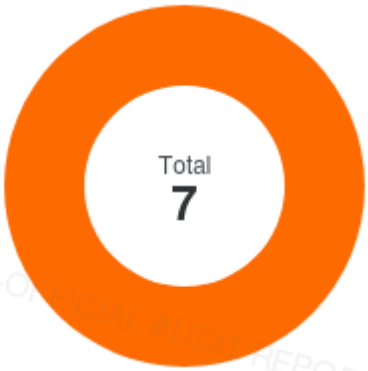
July 24, 2023

Executive Summary

Overview	
Project Name	69-2021-12-nftx (1OK-ANC) (1OK-FLB)
Codebase URL	https://github.com/code-423n4/2021-12-nftx
Scan Engine	AI Analyzer
Scan Time	2023/07/24 01:39:06
Commit Id	6bedf40

Total	
Critical Issues	0
High risk Issues	7
Medium risk Issues	0
Low risk Issues	0
Informational Issues	0

Critical Issues	The issue can cause large economic losses, large-scale data disorder, loss of control of authority management, failure of key functions, or indirectly affect the correct operation of other smart contracts interacting with it.
High Risk Issues	The issue puts a large number of users' sensitive information at risk or is reasonably likely to lead to catastrophic impacts on clients' reputations or serious financial implications for clients and users.
Medium Risk Issues	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk Issues	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
Informational Issue	The issue does not pose an immediate risk but is relevant to security best practices or Defence in Depth.



Critical Issues	0%	0
High risk Issues	100%	7
Medium risk Issues	0%	0
Low risk Issues	0%	0
Informational Issues	0%	0

Summary of Findings

MetaScan security assessment was performed on **July 24, 2023 01:39:06** on project **69-2021-12-nftx (1OK-ANC) (1OK-FLB)** with the repository **https://github.com/code-423n4/2021-12-nftx** on branch **default branch**. The assessment was carried out by scanning the project's codebase using the scan engine **AI Analyzer**. There are in total **7** vulnerabilities / security risks discovered during the scanning session, among which **0** critical vulnerabilities, **7** high risk vulnerabilities, **0** medium risk vulnerabilities, **0** low risk vulnerabilities, **0** informational issues.

ID	Description	Severity
MSA-001	MWE-203: Approval Not Revoked	High risk
MSA-002	MWE-203: Approval Not Revoked	High risk
MSA-003	MWE-202: Insecure Token Buying Behavior	High risk
MSA-004	MWE-202: Insecure Token Buying Behavior	High risk
MSA-005	MWE-203: Approval Not Revoked	High risk
MSA-006	MWE-203: Approval Not Revoked	High risk
MSA-007	MWE-203: Approval Not Revoked	High risk

Findings

Critical (0)

No Critical vulnerabilities found here

High risk (7)

1. MWE-203: Approval Not Revoked

High risk Security Analyzer

Approval is not revoked or reset after the code functionality finishes.

File(s) Affected

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #447-464

```
447   function _swap1155(
448     uint256 vaultId,
449     uint256[] memory idsIn,
450     uint256[] memory amounts,
451     uint256[] memory idsOut,
452     address to
453   ) internal returns (address) {
454     address vault = nftxFactory.vault(vaultId);
455     require(vault != address(0), "NFTXZap: Vault does not exist");
456
457     // Transfer tokens to zap and mint to NFTX.
458     address assetAddress = INFTXVault(vault).assetAddress();
459     IERC1155Upgradeable(assetAddress).safeBatchTransferFrom(msg.sender, address(this), idsIn, amounts,
460     IERC1155Upgradeable(assetAddress).setApprovalForAll(vault, true);
461     INFTXVault(vault).swapTo(idsIn, amounts, idsOut, to);
462
463     return (vault);
464   }
```

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #284-314

```
function buyAndSwap1155WETH(
  uint256 vaultId,
  uint256[] memory idsIn,
  uint256[] memory amounts,
  uint256[] memory specificIds,
  uint256 maxWethIn,
  address[] calldata path,
  address to
) public payable nonReentrant {
  require(to != address(0));
  require(idsIn.length != 0);
  IERC20Upgradeable(address(WETH)).transferFrom(msg.sender, address(this), maxWethIn);
  uint256 count;
  for (uint256 i = 0; i < idsIn.length; i++) {
      uint256 amount = amounts[i];
      require(amount > 0, "Transferring < 1");
      count += amount;
  }
  INFTXVault vault = INFTXVault(nftxFactory.vault(vaultId));
  uint256 redeemFees = (vault.targetSwapFee() * specificIds.length) + (
      vault.randomSwapFee() * (count - specificIds.length)
  );
  uint256[] memory swapAmounts = _buyVaultToken(address(vault), redeemFees, msg.value, path);
  _swap1155(vaultId, idsIn, amounts, specificIds, to);

  emit Swap(count, swapAmounts[0], to);

  // Return extras.
  uint256 remaining = WETH.balanceOf(address(this));
  WETH.transfer(to, remaining);
}
```

Recommendation

Try to remove the approval after the code finishes its job.

2. MWE-203: Approval Not Revoked

High risk Security Analyzer

Approval is not revoked or reset after the code functionality finishes.

File(s) Affected

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #477-494

```
477  function _mint1155(
478    uint256 vaultId,
479    uint256[] memory ids,
480    uint256[] memory amounts
481  ) internal returns (address, uint256) {
482    address vault = nftxFactory.vault(vaultId);
483    require(vault != address(0), "NFTXZap: Vault does not exist");
484
485    // Transfer tokens to zap and mint to NFTX.
486    address assetAddress = INFTXVault(vault).assetAddress();
487    IERC1155Upgradeable(assetAddress).safeBatchTransferFrom(msg.sender, address(this), ids, amounts, "
488    IERC1155Upgradeable(assetAddress).setApprovalForAll(vault, true);
489    uint256 count = INFTXVault(vault).mint(ids, amounts);
490    uint256 balance = (count * BASE) - INFTXVault(vault).mintFee()*count;
491    require(balance == IERC20Upgradeable(vault).balanceOf(address(this)), "Did not receive expected bal
492
493    return (vault, balance);
494  }
```

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #385-403

```
385  function mintAndSell1155WETH(
386    uint256 vaultId,
387    uint256[] memory ids,
388    uint256[] memory amounts,
389    uint256 minWethOut,
390    address[] calldata path,
391    address to
392  ) public nonReentrant {
393    require(to != address(0));
394    require(ids.length != 0);
395    (address vault, uint256 vaultTokenBalance) = _mint1155(vaultId, ids, amounts);
396    _sellVaultTokenWETH(vault, minWethOut, vaultTokenBalance, path, to);
397
398    uint256 count;
399    for (uint256 i = 0; i < ids.length; i++) {
400        count += amounts[i];
401    }
402    emit Sell(count, amounts[1], to);
403  }
```

Recommendation

Try to remove the approval after the code finishes its job.

3. MWE-202: Insecure Token Buying Behavior

High risk Security Analyzer

Buying tokens via swap or AMM can be manipulated to cause sandwich attacks.

File(s) Affected

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #496-511

```
496  function _buyVaultToken(
497    address vault,
498    uint256 minTokenOut,
499    uint256 maxWethIn,
500    address[] calldata path
501  ) internal returns (uint256[] memory) {
502    uint256[] memory amounts = sushiRouter.swapTokensForExactTokens(
503      minTokenOut,
504      maxWethIn,
505      path,
506      address(this),
507      block.timestamp
508    );
509
510    return amounts;
511  }
```

Recommendation

Do not use swap or AMM to buy tokens.

4. MWE-202: Insecure Token Buying Behavior

High risk Security Analyzer

Buying tokens via swap or AMM can be manipulated to cause sandwich attacks.

File(s) Affected

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #496-511

```
function _buyVaultToken(
  address vault,
  uint256 minTokenOut,
  uint256 maxWethIn,
  address[] calldata path
) internal returns (uint256[] memory) {
  uint256[] memory amounts = sushiRouter.swapTokensForExactTokens(
    minTokenOut,
    maxWethIn,
    path,
    address(this),
    block.timestamp
  );

  return amounts;
}
```

nftx-protocol-v2/contracts/solidity/NFTXMarketplaceZap.sol #341-363

```
function buyAndRedeemWETH(
  uint256 vaultId,
  uint256 amount,
  uint256[] memory specificIds,
  uint256 maxWethIn,
  address[] calldata path,
  address to
) public nonReentrant {
  require(to != address(0));
  require(amount != 0);
  IERC20Upgradeable(address(WETH)).transferFrom(msg.sender, address(this), maxWethIn);
  INFTXVault vault = INFTXVault(nftxFactory.vault(vaultId));
  uint256 totalFee = (vault.targetRedeemFee() * specificIds.length) + (
      vault.randomRedeemFee() * (amount - specificIds.length)
  );
  uint256[] memory amounts = _buyVaultToken(address(vault), (amount*BASE) + totalFee, maxWethIn, path
  _redeem(vaultId, amount, specificIds, to);

  emit Buy(amount, amounts[0], to);

  uint256 remaining = WETH.balanceOf(address(this));
  WETH.transfer(to, remaining);
}
```

Recommendation

Do not use swap or AMM to buy tokens.

5. MWE-203: Approval Not Revoked

High risk Security Analyzer

Approval is not revoked or reset after the code functionality finishes.

File(s) Affected

nftx-protocol-v2/contracts/solidity/NFTXStakingZap.sol #353-373

```
353  function _addLiquidity1155WETH(
354    uint256 vaultId,
355    uint256[] memory ids,
356    uint256[] memory amounts,
357    uint256 minWethIn,
358    uint256 wethIn,
359    address to
360  ) internal returns (uint256, uint256, uint256) {
361    address vault = nftxFactory.vault(vaultId);
362    require(vault != address(0), "NFTXZap: Vault does not exist");
363
364    // Transfer tokens to zap and mint to NFTX.
365    address assetAddress = INFTXVault(vault).assetAddress();
366    IERC1155Upgradeable(assetAddress).safeBatchTransferFrom(msg.sender, address(this), ids, amounts, "
367    IERC1155Upgradeable(assetAddress).setApprovalForAll(vault, true);
368    uint256 count = INFTXVault(vault).mint(ids, amounts);
369    uint256 balance = (count * BASE); // We should not be experiencing fees.
370    require(balance == IERC20Upgradeable(vault).balanceOf(address(this)), "Did not receive expected bal
371
372    return _addLiquidityAndLock(vaultId, vault, balance, minWethIn, wethIn, to);
373  }
```

nftx-protocol-v2/contracts/solidity/NFTXStakingZap.sol #253-271

```
253  function addLiquidity1155ETHTo(
254    uint256 vaultId,
255    uint256[] memory ids,
256    uint256[] memory amounts,
257    uint256 minEthIn,
258    address to
259  ) public payable nonReentrant returns (uint256) {
260    WETH.deposit{value: msg.value}();
261    // Finish this.
262    (, uint256 amountEth, uint256 liquidity) = _addLiquidity1155WETH(vaultId, ids, amounts, minEthIn, r
263
264    // Return extras.
265    if (amountEth < msg.value) {
266      WETH.withdraw(msg.value-amountEth);
267      payable(to).call{value: msg.value-amountEth};
268    }
269
270    return liquidity;
271  }
```

Recommendation

Try to remove the approval after the code finishes its job.

6. MWE-203: Approval Not Revoked

High risk Security Analyzer

Approval is not revoked or reset after the code functionality finishes.

File(s) Affected

nftx-protocol-v2/contracts/solidity/other/PalmNFTXStakingZap.sol #268-288

```
function _addLiquidity1155WETH(
  uint256 vaultId,
  uint256[] memory ids,
  uint256[] memory amounts,
  uint256 minWethIn,
  uint256 wethIn,
  address to
) internal returns (uint256, uint256, uint256) {
  address vault = nftxFactory.vault(vaultId);
  require(vault != address(0), "NFTXZap: Vault does not exist");

  // Transfer tokens to zap and mint to NFTX.
  address assetAddress = INFTXVault(vault).assetAddress();
  IERC1155Upgradeable(assetAddress).safeBatchTransferFrom(msg.sender, address(this), ids, amounts, "'
  IERC1155Upgradeable(assetAddress).setApprovalForAll(vault, true);
  uint256 count = INFTXVault(vault).mint(ids, amounts);
  uint256 balance = (count * BASE); // We should not be experiencing fees.
  require(balance == IERC20Upgradeable(vault).balanceOf(address(this)), "Did not receive expected bal

  return _addLiquidityAndLock(vaultId, vault, balance, minWethIn, wethIn, to);
}
```

nftx-protocol-v2/contracts/solidity/NFTXStakingZap.sol #253-271

```
function addLiquidity1155ETHTo(
  uint256 vaultId,
  uint256[] memory ids,
  uint256[] memory amounts,
  uint256 minEthIn,
  address to
) public payable nonReentrant returns (uint256) {
  WETH.deposit{value: msg.value}();
  // Finish this.
  (, uint256 amountEth, uint256 liquidity) = _addLiquidity1155WETH(vaultId, ids, amounts, minEthIn, r

  // Return extras.
  if (amountEth < msg.value) {
    WETH.withdraw(msg.value-amountEth);
    payable(to).call{value: msg.value-amountEth};
  }

  return liquidity;
}
```

Recommendation

Try to remove the approval after the code finishes its job.

7. MWE-203: Approval Not Revoked

High risk Security Analyzer

Approval is not revoked or reset after the code functionality finishes.

File(s) Affected

nftx-protocol-v2/contracts/solidity/eligibility/NFTXMintRequestEligibility.sol #203-232

```
203     function reclaimRequestedMint(uint256[] calldata tokenIds)
204         external
205         virtual
206     {
207         address _assetAddress = vault.assetAddress();
208         bool _is1155 = is1155;
209         for (uint256 i = 0; i < tokenIds.length; i++) {
210             uint256 tokenId = tokenIds[i];
211             uint256 amount = mintRequests[msg.sender][tokenId];
212             require(amount > 0, "NFTXVault: nothing to reclaim");
213             require(!approvedMints[msg.sender][tokenId], "Eligibility: cannot be approved");
214             mintRequests[msg.sender][tokenId] = 0;
215             approvedMints[msg.sender][tokenId] = false;
216             if (_is1155) {
217                 IERC1155Upgradeable(_assetAddress).safeTransferFrom(
218                     address(this),
219                     msg.sender,
220                     tokenId,
221                     amount,
222                     ""
223                 );
224             } else {
225                 IERC721(_assetAddress).safeTransferFrom(
226                     address(this),
227                     msg.sender,
228                     tokenId
229                 );
230             }
231         }
232     }
```

Recommendation

Try to remove the approval after the code finishes its job.

Medium risk (0)

No Medium risk vulnerabilities found here

Low risk (0)

No Low risk vulnerabilities found here

Informational (0)

No Informational vulnerabilities found here

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without MetaTrust's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts MetaTrust to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. MetaTrust's position is that each company and individual are responsible for their own due diligence and continuous security. MetaTrust's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by MetaTrust is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS 69-2021-12-nftx (1OK-ANC) (1OK-FLB) Security Assessment AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, MetaTrust

HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, MetaTrust SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, MetaTrust MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, MetaTrust PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER MetaTrust NOR ANY OF MetaTrust'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. MetaTrust WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT MetaTrust'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING 69-2021-12-nftx (1OK-ANC) (1OK-FLB) Security Assessment MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST MetaTrust WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF MetaTrust CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST MetaTrust WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.