CERTIK

Security Assessment

Nf3x - audit

CertiK Assessed on Dec 30th, 2022

CertiK Assessed on Dec 30th, 2022

Nf3x - audit

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Ethereum (ETH)	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Solidity	Delivered on 12/30/2022	N/A

CODEBASE

https://github.com/NF3Labs/contracts-V2/tree/25887407c49c4ef65732f2807bff032614457957

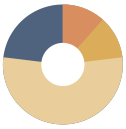
...View All

COMMITS

25887407c49c4ef65732f2807bff032614457957

...View All

Vulnerability Summary



26	19	0	0	7	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined

0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
3	Major	1 Resolved, 2 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
3	Medium	1 Resolved, 2 Acknowledged	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
14	Minor	13 Resolved, 1 Acknowledged	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
6	Informational	4 Resolved, 2 Acknowledged	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | NF3X - AUDIT

CODEBASE | NF3X - AUDIT

Repository

https://github.com/NF3Labs/contracts-V2/tree/25887407c49c4ef65732f2807bff032614457957

Commit

25887407c49c4ef65732f2807bff032614457957

AUDIT SCOPE | NF3X - AUDIT

20 files audited ● 10 files with Acknowledged findings ● 1 file with Resolved findings ● 9 files without findings

ID	File	SHA256 Checksum
NFT	contracts/tokens/NF3LoanPromissoryToken.sol	f69c62c0c26f2c295784d372016887f736159eb2ab827f081e6896387337697c
PTV	contracts/tokens/PositionToken.sol	3c4fb11ddfda382962e769f68a7c2c0dd23e786a99fc5d67049a4a3d2d146c1f
NFG	contracts/NF3GatedSwap.sol	a5408af80e55604c000885ba4326b90defa9805a441cc36ac4f52de513177339
NFL	contracts/NF3Loan.sol	00310e0a865c5d64ddb9a94907c035385380dab23f21a17b0d88724b36da27aa
NFM	contracts/NF3Market.sol	30b546fa6e127021714d02ea5bade3e030ef75a3bb82b2a942098aadd348cf38
NFP	contracts/NF3Proxy.sol	2340bb6a63824c07091c3c810b1c046eb7a9da8620c6d3effa8ea34dc4d7eb4c
RVN	contracts/Reserve.sol	2ac2ea18b5fd65e0b8f9da60798d0b4b8a86799c68463e9d8d99f8f7b643118c
SVN	contracts/Swap.sol	6a5171e01c17cb55765791b3949f98c593aa7a0a4dd50388aa5840ca951ba94c
VVN	contracts/Vault.sol	476387852ff30bbd9cd4630ee68242a614688125357278033b1583f91e4eda58
WVN	contracts/Whitelist.sol	cbd9bf33ff4d1498bc5fa9a7f27ef28d189c2a5119426fe7a490f51a112dd668
DTV	utils/DataTypes.sol	174581e7440da259e25135fd2e1e93572654f80054e6f57586e25a85f1f390d9
LDT	utils/LoanDataTypes.sol	37e38f496557e61420ee5d9a2eda26d1d2f9c1f0f19632d95eec12ea1595586f
INF	contracts/Interfaces/INF3Loan.sol	8e6c4dae6b1ea28a2b5c9925a23f87114d349ccdc8686bde5acfef7d349e245c
INM	contracts/Interfaces/INF3Market.sol	6989de2065192c86fa3adc9de6148a6726f5f12bb0e1f47e2dfcf7c6a3a4f5de

ID	File	SHA256 Checksum
IPT	contracts/Interfaces/IPositionToken.sol	979b3e883f17b328e77ba47b0d74cd168e681f1bc1568a607b17a695e49e545a
IRI	contracts/Interfaces/IReserve.sol	1c009538ae05e00eeac86015a413943ef23657f3b89d27d52050492f1b10e053
ISI	contracts/Interfaces/ISwap.sol	4ab792a24df96c3d4f3b795aefab8952eccc4d802df547589c8c145879e1c64f
IVI	contracts/Interfaces/IVault.sol	f885a99aa46e3b756595ac724cae20adb47571a6f914e65df2d8a5d219df6729
IWI	contracts/Interfaces/IWhitelist.sol	be5a657a376f74765536969069c8823f889ddfbd7a5f8dfc4713c411cd1ea0a2
UVN	contracts/lib/Utils.sol	36bb2817943045b28659facc9df076d5c2de3054cf85bea113cc5dba4bc559b8

APPROACH & METHODS | NF3X - AUDIT

This report has been prepared for Nf3x to discover issues and vulnerabilities in the source code of the Nf3x - audit project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis techniques.

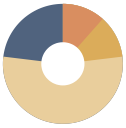
The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

FINDINGS | NF3X - AUDIT



26	0	3	3	14	6
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Nf3x - audit. Through this audit, we have uncovered 26 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
VNF-01	**Centralized Control Of Contract Upgrade**	**Centralization / Privilege**	**Major**	**Acknowledged**
VNF-02	**Centralization Related Risks**	**Centralization / Privilege**	**Major**	**Acknowledged**
VVN-01	**`owner` Can Transfer Assets Users Approved For The Vault**	**Centralization / Privilege**	**Major**	**Resolved**
VNF-03	Lack Of Storage Gap	Language Specific	Medium	Resolved
VNF-04	Unknown `trustedForwarder` And Signature Verification	Volatile Code, Language Specific	Medium	Acknowledged
VVN-02	Incompatibility With Deflationary Tokens	Volatile Code	Medium	Acknowledged
DTV-01	Unused `RESERVED` Status	Logical Issue	Minor	Resolved
GLOBAL-01	Third Party Dependencies	Volatile Code	Minor	Acknowledged
NFG-01	Lack Of Sanity Check	Logical Issue	Minor	Resolved
NFL-01	State Variables In Upgradeable Contracts Are Initialized When Declared	Logical Issue	Minor	Resolved

ID	Title	Category	Severity	Status
NFL-02	Overdue Loan Can Payback	Logical Issue	Minor	Resolved
NFL-03	Lack Of Reasonable Fee Limitation	Logical Issue	Minor	Resolved
NFL-05	Function `updateLoanTerms()` Missing Nonce Invalidation	Logical Issue	Minor	Resolved
NFP-01	Potential Function Selector Clash Risk	Logical Issue, Language Specific	Minor	Resolved
PTV-01	Missing IPFS Check	Logical Issue	Minor	Resolved
PTV-02	Function `setTokenURI()` Concatenates Strings Incorrectly	Language Specific	Minor	Resolved
RVN-01	Lack Of Sanity Check On Reserve Duration	Logical Issue	Minor	Resolved
VNF-05	Not All Parent Contract Initializing Functions Are Called	Language Specific	Minor	Resolved
VNF-06	Missing Zero Address Validation	Volatile Code	Minor	Resolved
VNF-07	Uninitialized Logic Contract	Logical Issue	Minor	Resolved
NFG-02	Buy NF3 Banner NFT With Buyer Deployed Token	Logical Issue	Informational	Resolved
NFG-03	Eligibility Token Can Be Reused	Logical Issue	Informational	Acknowledged
NFM-01	Arbitrary Royalty Fee	Logical Issue	Informational	Acknowledged
NFP-02	Missing Error Messages	Coding Style	Informational	Resolved
NFT-01	Delete Non-Existent `loanDataHash`	Logical Issue	Informational	Resolved

ID	Title	Category	Severity	Status
VNF-08	Missing Emit Events	Coding Style	Informational	Resolved

VNF-01 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization / Privilege	**Major**	**contracts/NF3Loan.sol: 19; contracts/NF3Market.sol: 19; contracts/tokens/NF3LoanPromissoryToken.sol: 14**	**Acknowledged**

Description

`NF3Loan`, `NF3Market` and `NF3LoanPromissoryToken` are upgradeable contracts, the owner can upgrade the contract without the community's commitment. If an attacker compromises the account, he can change the implementation of the contract and drain tokens from the contract.

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND

- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

[Nf3x Team]:

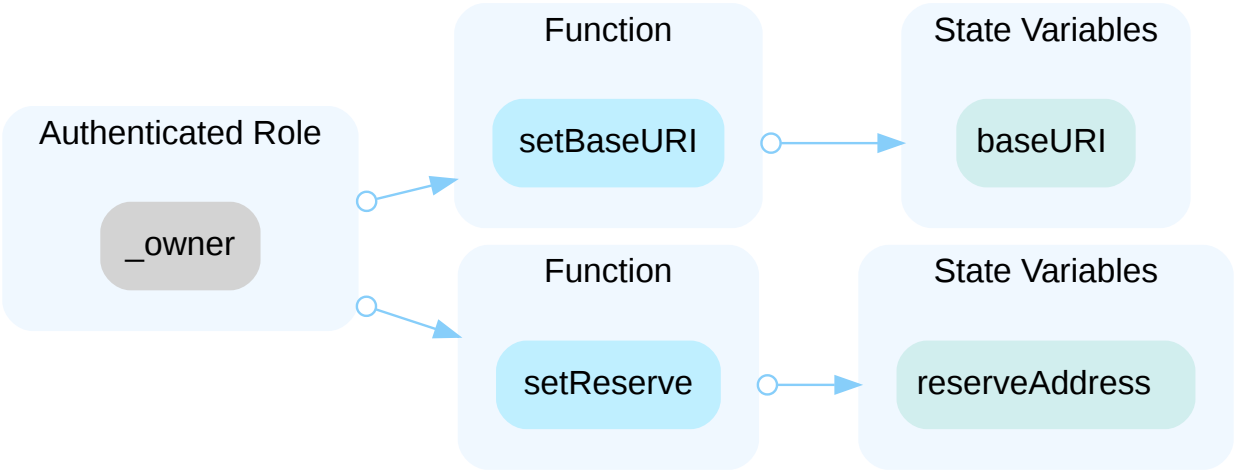
We will use the Gnosis Safe contract plus timelock.

VNF-02 | CENTRALIZATION RELATED RISKS

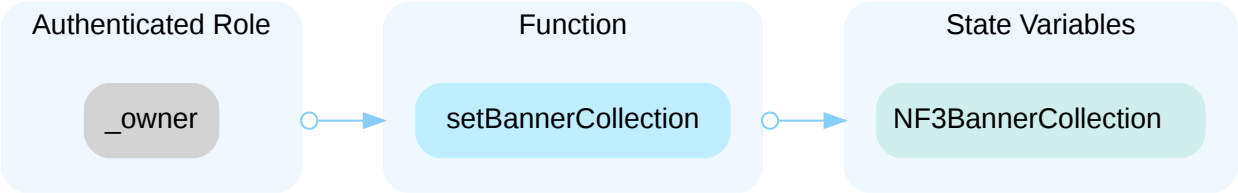
Category	Severity	Location	Status
Centralization / Privilege	Major	contracts/NF3GatedSwap.sol: 184; contracts/NF3Loan.sol: 426, 437, 447, 461; contracts/NF3Market.sol: 361, 368, 375, 382; contracts/NF3Proxy.sol: 45, 52; contracts/Reserve.sol: 479, 486, 493, 500, 511; contracts/Swap.sol: 388, 395, 402; contracts/Vault.sol: 70, 280, 287, 294, 304; contracts/Whitelist.sol: 86; contracts/tokens/PositionToken.sol: 152, 159	Acknowledged

Description

In the contract `PositionToken` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and set reserve address and base URI.



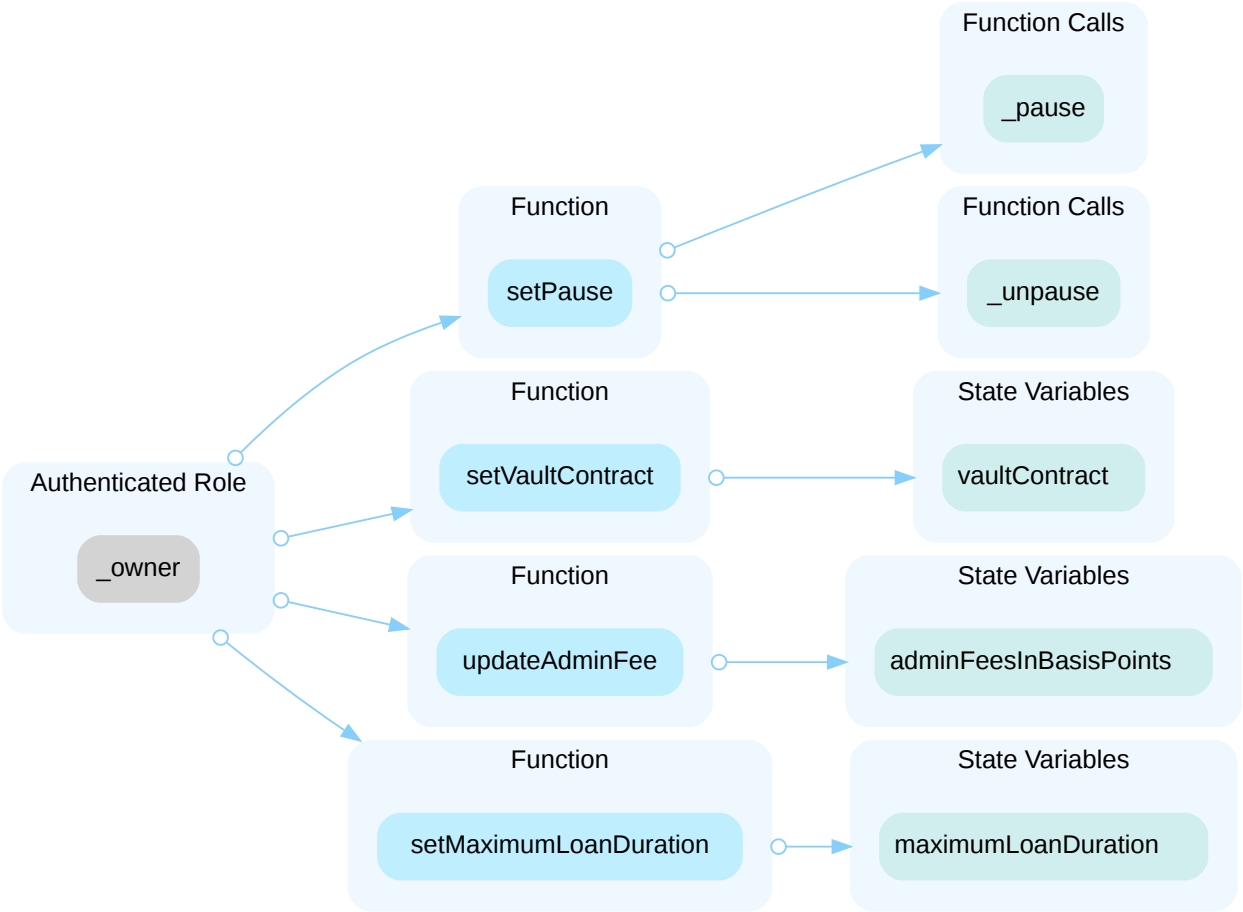
In the contract `NF3GatedSwap` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and set the banner collection address.



In the contract `NF3Loan` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and

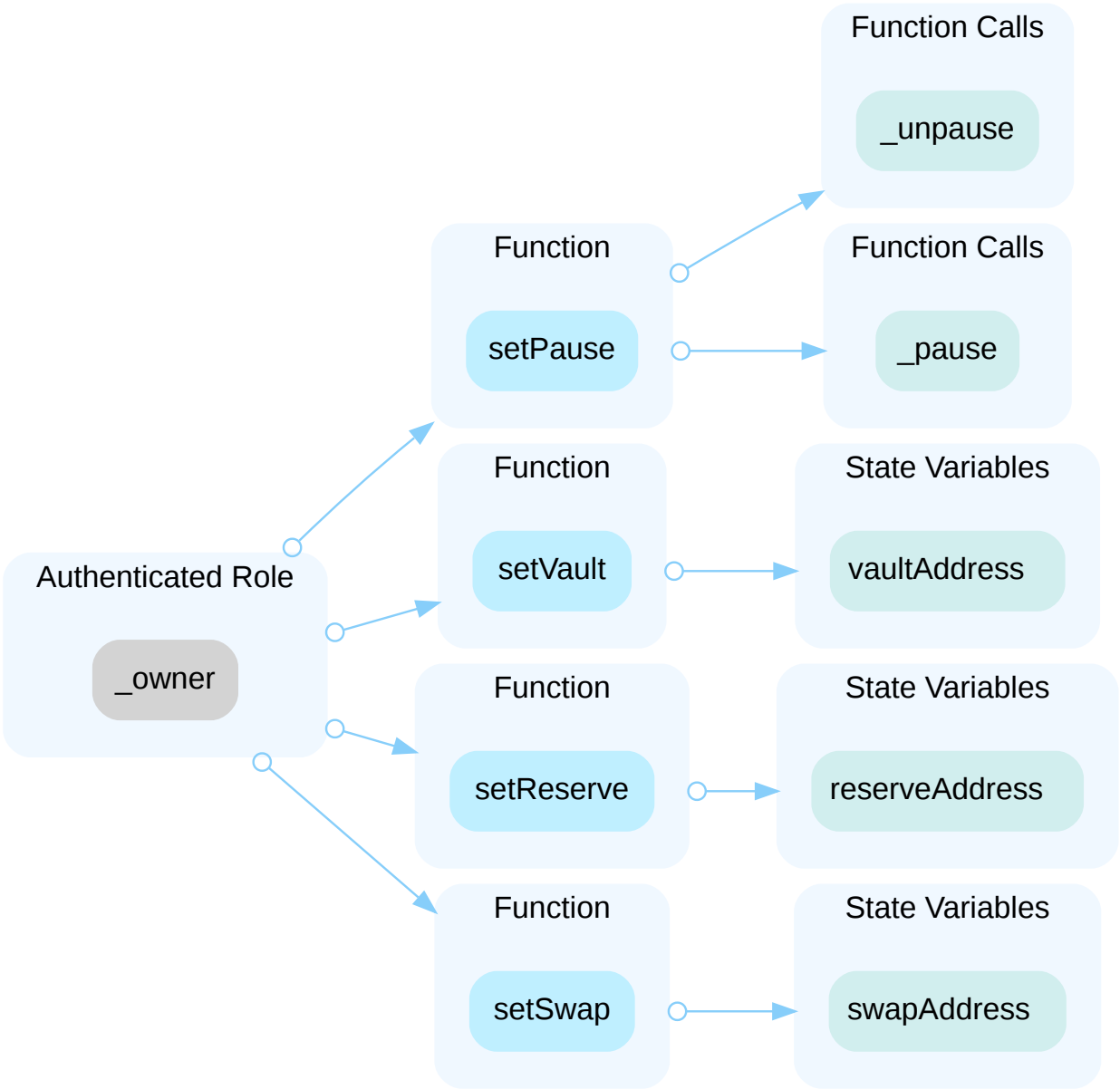
- set vault contract address
- set maximum loan duration
- set admin fees in basis points

- pause/unpause the contract



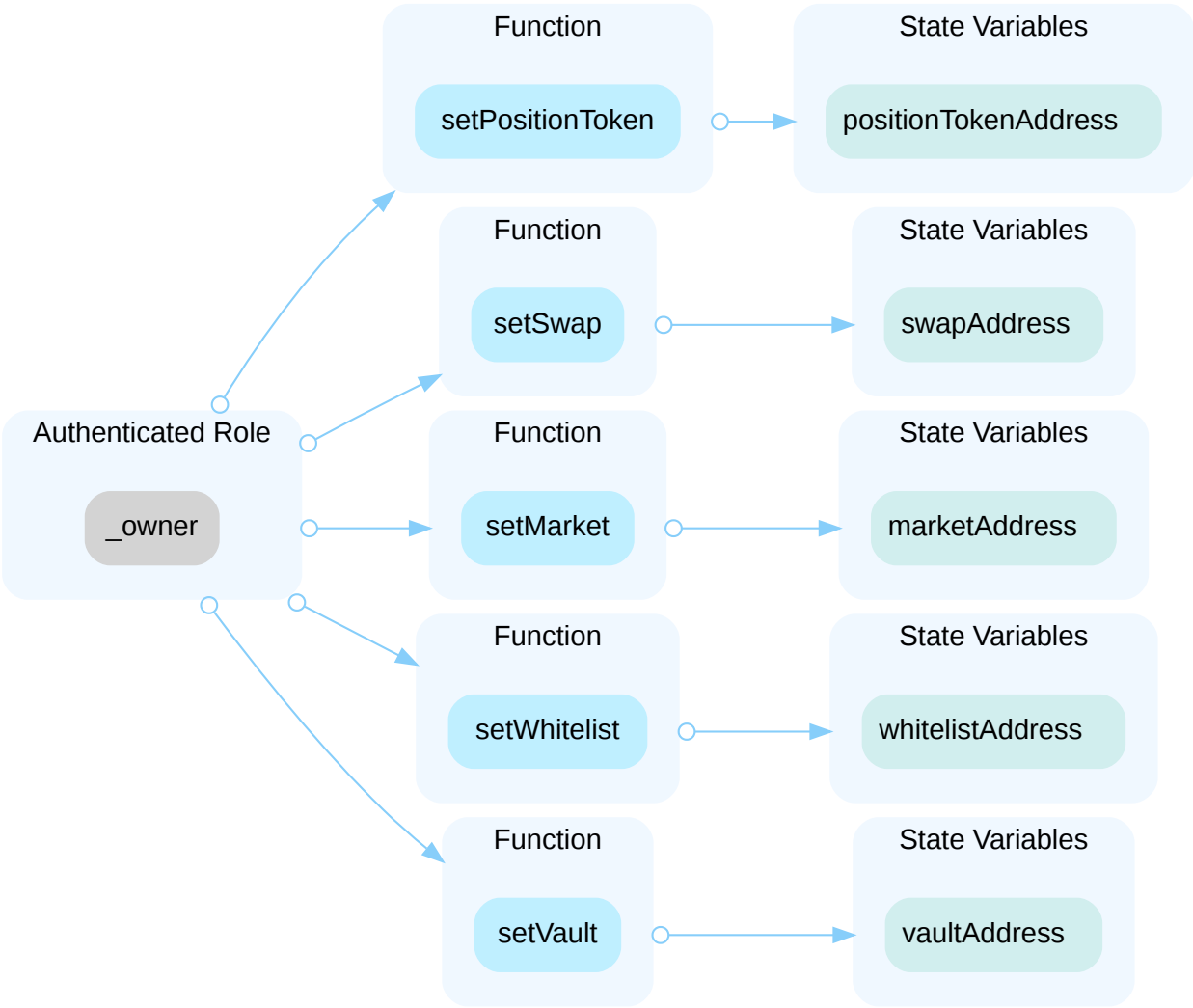
In the contract `NF3Market` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and

- set vault contract address
- set reserve contract address
- set swap contract address
- pause/unpause the contract



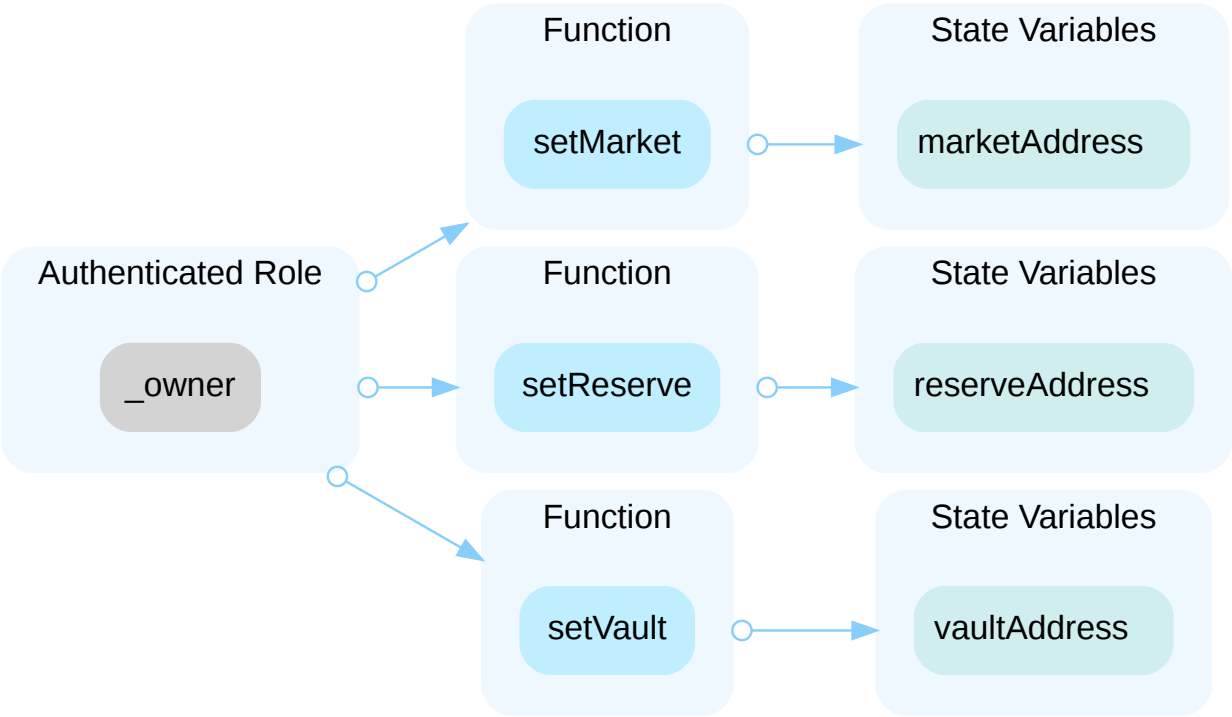
In the contract `Reserve` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and

- set vault contract address
- set whitelist contract address
- set swap contract address
- set market contract address
- set position token address



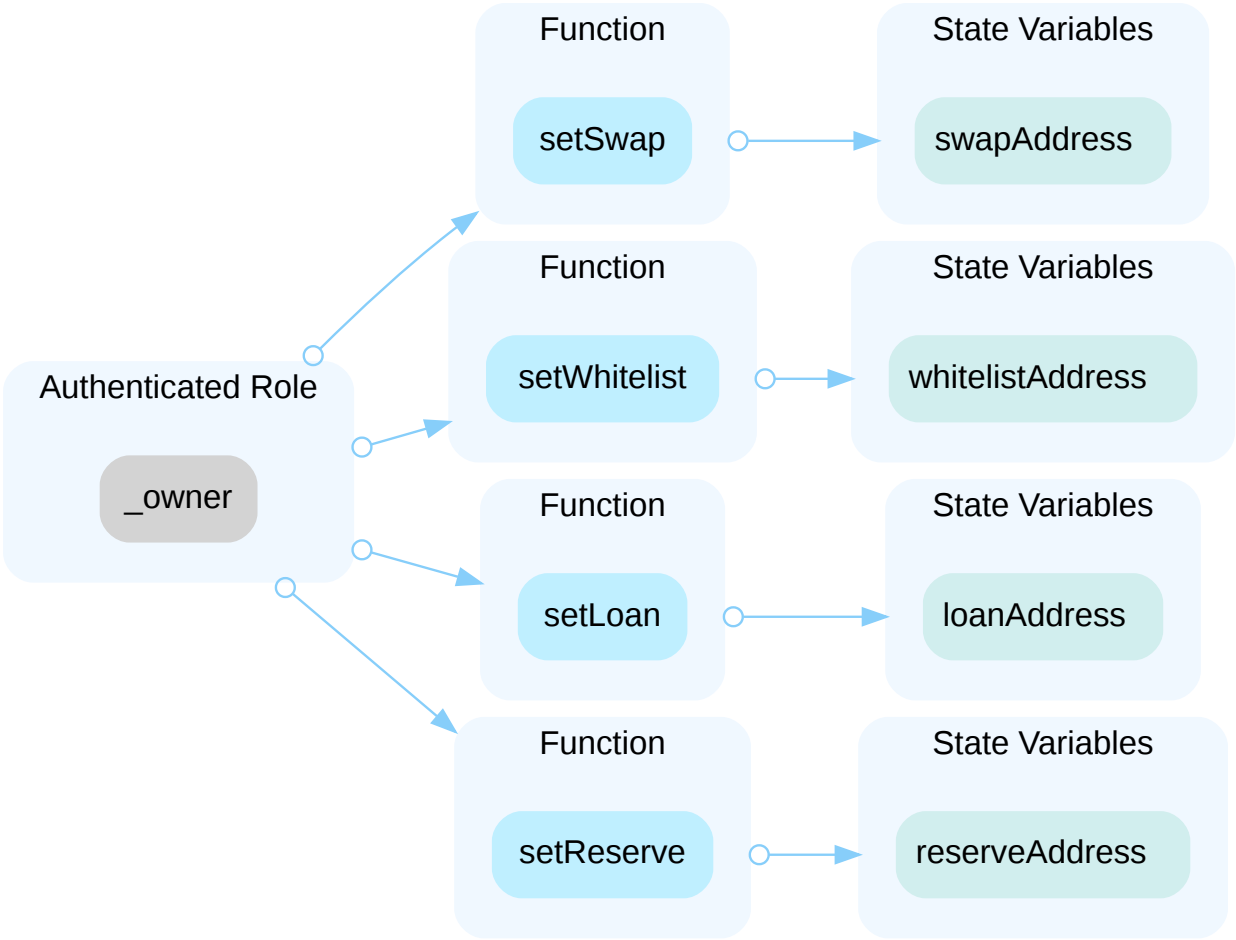
In the contract `Swap` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and

- set vault contract address
- set reserve contract address
- set market contract address

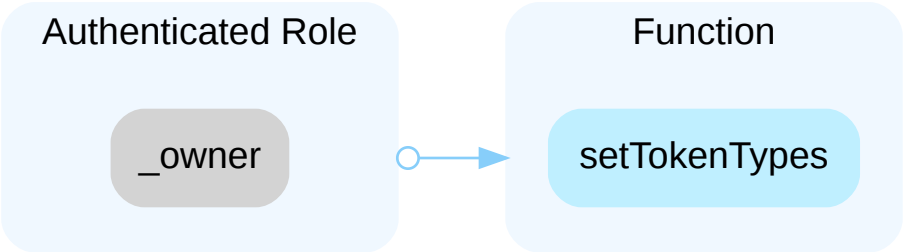


In the contract `Vault` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and

- set whitelist contract address
- set swap contract address
- set loan contract address
- set reserve address
- transfer all the assets that users have approved or reserved to the `Vault` contract



In the contract `Whitelist` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and set tokens type



In the contract `NF3Proxy` the role `proxyOwner` has authority over the functions

- transferProxyOwnership()
- upgradeTo() Any compromise to the `proxyOwner` account may allow the hacker to take advantage of this authority and
- transfer the proxy owner
- allow the proxy owner to upgrade the implementation contract

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We recommend carefully managing the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement;
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles;
  OR
- Remove the risky functionality.

Alleviation

[Nf3x Team]:

We will use the Gnosis Safe contract plus timelock.

VVN-01 `owner` CAN TRANSFER ASSETS USERS APPROVED FOR THE VAULT

Category	Severity	Location	Status
Centralization / Privilege	**Major**	**contracts/Vault.sol: 49, 64, 149, 206**	**Resolved**

Description

The `transferAssets()`, `receiveAssets()`, and `sendAssets()` functions all utilize the `safeTransferFrom()` function which require users to approve the contract. They are intended to be called by the other contracts in the code base that contain relevant checks. However, the `onlyApproved()` modifier in the `Vault` contract would pass for the `owner` of the contract, enabling the owner to transfer the approved assets to owner designated addresses. If the `owner` address is compromised, all the assets that users have approved the `Vault` contract for could be drained.

Recommendation

We recommend removing `owner` from the `onlyApproved()` modifier.

Alleviation

The client revised the code and resolved the issue in this commit.

VNF-03 | LACK OF STORAGE GAP

Category	Severity	Location	Status
Language Specific	Medium	contracts/NF3Loan.sol: 19; contracts/NF3Market.sol: 19; contracts/tokens/NF3LoanPromissoryToken.sol: 14	Resolved

Description

The code base includes a transparent upgradeable proxy pattern, implying the potential to upgrade implementation contracts in the future.

For upgradeable contracts, there must be storage gap to "allow developers to freely add new state variables in the future without compromising the storage compatibility with existing deployments". Otherwise it may be very difficult to write new implementation code. Without storage gap, the variable in child contract might be overwritten by the upgraded base contract if new variables are added to the base contract.

Refer to https://docs.openzeppelin.com/upgrades-plugins/1.x/writing-upgradeable

Recommendation

We recommend having a storage gap of a reasonable size preserved in the logic contract in case that new state variables are introduced in future upgrades. For more information, please refer to: https://docs.openzeppelin.com/contracts/3.x/upgradeable#storage_gaps.

Alleviation

The team heeded the advice and resolved the finding in the commit.

VNF-04 UNKNOWN `trustedForwarder` AND SIGNATURE VERIFICATION

Category	Severity	Location	Status
Volatile Code, Language Specific	Medium	contracts/NF3Loan.sol: 62, 685~701; contracts/NF3Market.sol: 46, 406~422	Acknowledged

Description

Both the `NF3Market` and the `NF3Loan` contracts extend ERC2771 and utilize a `trustedForwarder` address for meta transactions. The audit scope does not contain `trustedForwarder`, and its signature verification scheme is unknown. If the signature verification is implemented incorrectly, the `trustedForwarder` contract could potentially execute transaction on behalf of other users without their consent.

Recommendation

The team should make every effort to ensure the functional correctness of out-of-scope contracts.

Alleviation

[Nf3x Team]:

The trusted forwarder is not yet implemented and is there for future improvements. We will use a placeholder proxy contract right now and upgrade it later. Hence currently out of scope.

VVN-02 | INCOMPATIBILITY WITH DEFLATIONARY TOKENS

Category	Severity	Location	Status
Volatile Code	Medium	contracts/Vault.sol: 190~194, 250~253, 459~463	Acknowledged

Description

When transferring standard ERC20 deflationary tokens, the input amount may not be equal to the received amount due to the charged transaction fee. As a result, an inconsistency in the amount will occur and the transaction may fail due to the validation checks.

Scenario

1. The seller reserves 100 deflationary tokens (with a 10% transaction fee) to the vault contract, only 90 tokens actually arrive to the contract.
2. The buyer calls `payRemains()` function to pay the remaining, but the token balance in the vault contract is insufficient, causing the reserve swap to fail.

Recommendation

We recommend regulating the set of payment tokens supported and adding necessary mitigation mechanisms to keep track of accurate balances if there is a need to support deflationary tokens.

Alleviation

[Nf3x Team]:

Current implementation does not support deflationary tokens and such tokens will not be whitelisted on the platform for use until the contracts have support for them.

DTV-01 | UNUSED `RESERVED` STATUS

Category	Severity	Location	Status
Logical Issue	Minor	utils/DataTypes.sol: 164	Resolved

Description

From the NF3 function flows document, we understand that the nonce of the listing is set as `RESERVED` in the functions `reserveDeposit()` and `acceptListedReserveOffer()`. After calling the function `payRemains()` or `claimDefaulted()`, the nonce of the listing is set to `EXHAUSTED`. However, the functions `reserveDeposit()` and `acceptListedReserveOffer()` in the contract `Reserve` directly set the state as `EXHAUSTED`, lacking the transition process of `RESERVED`. Inconsistency with the implementation mentioned in the documentation.

Recommendation

We would like to confirm with the client if the current implementation aligns with the intended project design.

Alleviation

The client revised the code and resolved the issue in this commit.

GLOBAL-01 | THIRD PARTY DEPENDENCIES

Category	Severity	Location	Status
Volatile Code	Minor		Acknowledged

Description

The contracts in the code base interact with one or more third party protocols, including but not limited to external ERC20, ERC721, ERC1155, crypto Kitties and punks contracts, trusted forwarder, etc. The scope of the audit treats third party entities as black boxes and assume their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades or reconfiguration of third parties can possibly create severe impacts, such as increasing fees of third parties, transfer blacklists, etc.

Recommendation

We understand that the business logic requires interaction with the third parties. We encourage the team to constantly monitor the statuses of third parties to mitigate the side effects when unexpected activities are observed.

Alleviation

[Nf3x Team]:

We need to interact with third party contracts for the project logic. We will constantly monitor status of such contracts and take necessary actions whenever required.

NFG-01 | LACK OF SANITY CHECK

Category	Severity	Location	Status
Logical Issue	Minor	contracts/NF3GatedSwap.sol: 178	Resolved

Description

The function `cancelNF3GatedListing()` lacks of a sanity check to ensure that the sold NFT number is less than `_listing.editions`, which causes sold listings to still be cancelled and emit a misleading `NF3GatedListingCancelled` event.

Recommendation

We recommend adding the aforementioned check.

Alleviation

The client revised the code and resolved the issue in this commit.

NFL-01 STATE VARIABLES IN UPGRADEABLE CONTRACTS ARE INITIALIZED WHEN DECLARED

Category	Severity	Location	Status
Logical Issue	Minor	contracts/NF3Loan.sol: 51	Resolved

Description

State variables initialized when declared are equivalent to setting them inside the constructor. Therefore, setting state variables when declared in a logic contract has no actual effect since the constructor in the logic contract does not affect the storage variable in the proxy contract.

Recommendation

We recommend initializing state variables in an initializer function if necessary to avoid unexpected behavior and confusion.

Alleviation

The client revised the code and resolved the issue in this commit.

NFL-02 | OVERDUE LOAN CAN PAYBACK

Category	Severity	Location	Status
Logical Issue	Minor	contracts/NF3Loan.sol: 237	Resolved

Description

In the `payBackLoan()` function , there is no check to see if the loan is overdue. When the loan time period expires, the borrower can call the `payBackLoan()` function to repay the loan and get back the collateral before the lender calls the `claimOverdueLoanCollateral()` function , and the lender is not eligible to seize the collateral.

Scenario

1. The lender creats a listing, that includes all the terms he is ready to loan his NFT for
2. The borrower accepts this listing and starts the loan through the `beginLoan()` funtion
3. When the loan time period expires, the borrower pays back the loan amount + interest accumulated through the `payBackLoan()` function, resulting in the lender not being eligible to seize the collateral through the `claimOverdueLoanCollateral()` function.

Recommendation

We advise the client to add a check that the loan is not overdue.

Alleviation

[Nf3x Team]:

This is a project decision where the borrower is allowed to payback the loan until lender has not claimed the collateral.

NFL-03 | LACK OF REASONABLE FEE LIMITATION

Category	Severity	Location	Status
Logical Issue	Minor	contracts/NF3Loan.sol: 452	Resolved

Description

The `_adminFeeInBasisPoints` can be set up to 100%.

```
452         if (_adminFeeInBasisPoints >= 10000) {
453             revert NF3LoanError(NF3LoanErrorCodes.INVALID_BASIS_VALUE);
454         }
```

Recommendation

We recommend setting a reasonable upper limit for fees.

Alleviation

The client revised the code and resolved the issue in this commit.

NFL-05 FUNCTION `updateLoanTerms()` MISSING NONCE INVALIDATION

Category	Severity	Location	Status
Logical Issue	Minor	contracts/NF3Loan.sol: 325	Resolved

Description

The `updateLoanTerms()` function does not invalidate the nonce, resulting in the signature of the loan offer potentially being reused.

Recommendation

We recommend invalidating the nonce when it is used.

Alleviation

The client fixed it in commit da1306f256d512c73654d8497d6ab57e10cab5c7.

NFP-01 | POTENTIAL FUNCTION SELECTOR CLASH RISK

Category	Severity	Location	Status
Logical Issue, Language Specific	Minor	contracts/NF3Proxy.sol: 45, 52	Resolved

Description

The owner of the proxy can call the `upgradeTo()` function to upgrade the proxy to a new logic contract. The `NF3Proxy` does not prevent delegating owner's function call to the logic contract. If the logic contract has a function with the same name, calling `upgradeTo()` function may lead to unintended errors, or even malicious exploits.

Furthermore, it is possible that the logic contract has a function with different names but have the same function selector as the proxy's `upgradeTo()` function. This could cause the owner to inadvertently upgrade a proxy to a random address while attempting to call a completely different function provided by the implementation.

Refer to https://blog.openzeppelin.com/the-transparent-proxy-pattern/

Recommendation

We recommend applying the transparent proxy pattern.

Alleviation

The team heeded the advice and resolved the finding in the commit.

PTV-01 | MISSING IPFS CHECK

Category	Severity	Location	Status
Logical Issue	Minor	contracts/tokens/PositionToken.sol: 162	Resolved

Description

The function doesn't verify if the supplied `_baseURI` is a valid IPFS location.

Recommendation

It's recommended to use IPFS for NFT token metadata rather than HTTP/HTTPS. See https://docs.openzeppelin.com/contracts/4.x/erc721.

Alleviation

[Nf3x Team]:

We will be storing position token metadata on the backend only. So seems doesn't need to check.

PTV-02 FUNCTION `setTokenURI()` CONCATENATES STRINGS INCORRECTLY

Category	Severity	Location	Status
Language Specific	Minor	contracts/tokens/PositionToken.sol: 172	Resolved

Description

The `setTokenURI()` function concatenates string `baseURI` with uint256 `_tokenId`, but this results in the `_tokenId` not being appended to the `baseURI` correctly.

Recommendation

We recommend converting `_tokenId` to a string before appending it to `baseURI`.

Alleviation

The client fixed it in commit 71d5488d9825576ba403e31f2a58bbd07458a898.

RVN-01 | LACK OF SANITY CHECK ON RESERVE DURATION

Category	Severity	Location	Status
Logical Issue	Minor	contracts/Reserve.sol: 151, 196, 255, 320	Resolved

Description

In a reservation swap, if the buyer does not pay the remaining in a given time frame `ReserveInfo.duration`, he forfeits his deposit. The `Reserve` contract lacks a sanity check to ensure the `ReserveInfo.duration` is a reasonable time for the buyer to pay the remaining balance.

Recommendation

We recommend adding a sanity check to ensure `ReserveInfo.duration` is reasonable.

Alleviation

The team heeded the advice and resolved the finding in the commit.

VNF-05 NOT ALL PARENT CONTRACT INITIALIZING FUNCTIONS ARE CALLED

Category	Severity	Location	Status
Language Specific	Minor	contracts/NF3Loan.sol: 66; contracts/NF3Market.sol: 50	Resolved

Description

Contract NF3Market/NF3Loan extends ReentrancyGuardUpgradeable, while `__ReentrancyGuard_init()` is not called in the initialize function. Generally, the initializer function of an upgradeable contract should always call all the initializer functions of the contracts that it extends.

Recommendation

We advise the client to call `__ReentrancyGuard_init()` in the linked contract.

Alleviation

The client revised the code and resolved the issue in this commit.

VNF-06 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	Minor	contracts/NF3GatedSwap.sol: 185; contracts/NF3Loan.sol: 433; contracts/NF3Market.sol: 364, 371, 378; contracts/Reserve.sol: 482, 489, 496, 507, 517; contracts/Swap.sol: 391, 398, 405; contracts/Vault.sol: 283, 290, 300, 306; contracts/tokens/PositionToken.sol: 155	Resolved

Description

Addresses should be checked before assignment or external call to make sure they are not zero addresses.

Recommendation

We advise adding a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

The client revised the code and resolved the issue in this commit.

VNF-07 | UNINITIALIZED LOGIC CONTRACT

Category	Severity	Location	Status
Logical Issue	Minor	contracts/NF3Loan.sol: 19; contracts/NF3Market.sol: 19	Resolved

Description

The `initialize()` function of the implementation contracts `NF3Market` and `NF3Loan` are not called when deployed. They can potentially be initialized by a malicious user, which may cause unintended consequences.

Recommendation

We recommend adding the following to the constructor of the contracts:

```
constructor() {
    _disableInitializers();
}
```

See https://github.com/OpenZeppelin/openzeppelin-contracts-upgradeable/blob/master/contracts/proxy/utils/Initializable.sol.

Alleviation

The client revised the code and resolved the issue in this commit.

NFG-02 | BUY NF3 BANNER NFT WITH BUYER DEPLOYED TOKEN

Category	Severity	Location	Status
Logical Issue	Informational	contracts/NF3GatedSwap.sol: 86~87	Resolved

Description

The function `gatedSwap()` does not check parameters `_considerationToken` and `_considerationTokenId`, so the buyer who has an eligibility token can deploy an ERC721 contract, mint for himself and swap for NF3 Banner NFT.

Recommendation

We would like to confirm with the client if the current implementation aligns with the original project design.

Alleviation

[Nf3x Team]:

This is a project decision where the user should be able to swap NF3 Banner NFT with any NFT if he is eligible, i.e., holds a particular NFT collection.

NFG-03 | ELIGIBILITY TOKEN CAN BE REUSED

Category	Severity	Location	Status
Logical Issue	Informational	contracts/NF3GatedSwap.sol: 113~118	Acknowledged

Description

```
114         if (
115             claimedNFTs[_listing.tokenId][_eligibilityToken][
116                 _eligibilityTokenId
117             ]
118         ) revert NF3GatedSwapError(NF3GatedSwapErrorCodes.NOT_ELIGIBLE);
```

The buyer's eligibility token can be reused multiple times as a pass token and use any NFT to swap for each tokenID of NF3 Banner NFT.

The lines L130-L132 mark eligibility token has been used for a token ID, for example, Banner NFT 1. A buyer who holds an eligibility token can still swap any NFT for Banner NFT with a tokenId of 2, 3, and so on.

Recommendation

We would like to confirm with the client if the current implementation aligns with the original project design.

Alleviation

[Nf3x Team]:

This is not an issue. Eligibility tokens mark eligibility for only a particular token ID. It should be reusable for other token IDs if set by the admin.

NFM-01 | ARBITRARY ROYALTY FEE

Category	Severity	Location	Status
Logical Issue	Informational	contracts/NF3Market.sol: 129, 159, 204, 258, 302, 326	Acknowledged

Description

Traders can specify any value for `royalty` parameter, allowing they to set any address as the royalty fee recipient and specify any fee rate. If the value of `royalty.percentage` is set to zero, the trader does not need to pay any fees. Furthermore, the `Vault` contract does not verify that the `royalty.to[0]` is the platform owner address.

Recommendation

We would like to confirm with the client if the current implementation aligns with the original project design.

Alleviation

[Nf3x Team]:

This is a project decision where the platform's frontend and user can choose to opt for royalty and it's percentage, i.e., it is not enforced on the smart contract level.

NFP-02 | MISSING ERROR MESSAGES

Category	Severity	Location	Status
Coding Style	Informational	contracts/NF3Proxy.sol: 46, 63, 65	Resolved

Description

The **require** can be used to check for conditions and throw an exception if the condition is not met. It is better to provide a string message containing details about the error that will be passed back to the caller.

Recommendation

We advise adding error messages to the linked **require** statements.

Alleviation

The client revised the code and resolved the issue in this commit.

NFT-01 | DELETE NON-EXISTENT `loanDataHash`

Category	Severity	Location	Status
Logical Issue	Informational	contracts/tokens/NF3LoanPromissoryToken.sol: 50	Resolved

Description

When the loan starts, both borrower and lender are issued a promissory token which can be further sold to as debt positions. The `loanId` is the same as the lender's promissory `tokenId` and the borrower's promissory `tokenId` is the lender's `tokenId + 1`. The key of the `loanDataHash` is the `loanId`. Therefore, when burning the promissory token of both lenders and borrowers, just delete the key as `_tokenId` because `_tokenId + 1` does not exist.

Recommendation

We advice the client to remove the redundant code.

Alleviation

The client revised the code and resolved the issue in this commit.

VNF-08 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	Informational	contracts/NF3GatedSwap.sol: 184; contracts/NF3Proxy.sol: 45, 52	Resolved

Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles.

Recommendation

It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

Alleviation

The client revised the code and resolved the issue in this commit.

APPENDIX | NF3X - AUDIT

Finding Categories

Categories	Description
Centralization / Privilege	Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.
Logical Issue	Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.
Language Specific	Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.
Coding Style	Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

DISCLAIMER | CERTIK

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 CertiK'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 CertiK 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. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK'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 CertiK 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 AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK 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, CERTIK 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, CERTIK 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 CERTIK NOR ANY OF CERTIK'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. CERTIK 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 CERTIK'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 MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK 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 CERTIK 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.

CertiK | Securing the Web3 World

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.