METATRUST

Security Assessment for

Inswap II

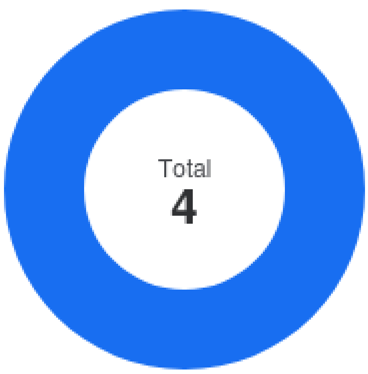
February 01, 2024

Executive Summary

Overview	
Project Name	Inswap II
Codebase URL	https://github.com/inswapio/inswap-contracts
Scan Engine	Security Analyzer
Scan Time	2024/02/01 08:00:00
Commit Id	d5145f381c56c47e365814ba8599e9c19f3dda4d

Total	
Critical Issues	0
High risk Issues	0
Medium risk Issues	0
Low risk Issues	0
Informational Issues	4

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



Critical Issues	0%	**0**
High risk Issues	0%	**0**
Medium risk Issues	0%	**0**
Low risk Issues	0%	**0**
Informational Issues	100%	**4**

Summary of Findings

MetaScan security assessment was performed on **February 01, 2024 08:00:00** on project **Inswap II** with the repository on branch **default branch**. The assessment was carried out by scanning the project's codebase using the scan engine **Security Analyzer**. There are in total **4** vulnerabilities / security risks discovered during the scanning session, among which **0** critical vulnerabilities, **0** high risk vulnerabilities, **0** medium risk vulnerabilities, **0** low risk vulnerabilities, **4** informational issues.

ID	Description	Severity	Alleviation
MSA-001	Redundant Check	Informational	Acknowledged
MSA-002	Missing Emitting Events	Informational	Acknowledged
MSA-003	Unused Private State Variable	Informational	Acknowledged
MSA-004	Checking on `registry` early	Informational	Acknowledged

Findings

Critical (0)

No Critical vulnerabilities found here

High risk (0)

No High risk vulnerabilities found here

Medium risk (0)

No Medium risk vulnerabilities found here

Low risk (0)

No Low risk vulnerabilities found here

Informational (4)

1. Redundant Check Informational Security Analyzer

Both the `homogenize` function and the `homogenizeMany` function validate the `unplugs[token]`, meanwhile, the `homogenizeMany` function invokes the `homogenize` function that already does the check on the `unplugs[token]`, thus, there is no need to repeatedly do it from the `homogenizeMany` function.

File(s) Affected

InsRegistry.sol #150-175

```
    function homogenize(address token, uint256 tokenId) public {
        require(unplugs[token], "Homogenization cannot be performed yet");
        Ins721 ins721 = Ins721(token);
        address owner = ins721.ownerOf(tokenId);
        require(owner == _msgSender(), "Not allow to homogenize");
        Ins20 ins20 = Ins20(bindings[address(ins721)]);
        ins20.mint(owner, ins721.toTick().lim);
        ins721.burn(tokenId);

        emit Inscribe(
            address(ins721),
            tokenId,
            address(0),
            string(ins721.applicationJson("burn"))
        );
    }

    function homogenizeMany(address token, uint256[] memory tokenIds) external {
        require(unplugs[token], "Homogenization cannot be performed yet");
        for (uint256 i = 0; i < tokenIds.length; ) {
            homogenize(token, tokenIds[i]);
            unchecked {
                ++i;
            }
        }
    }
```

Recommendation

Recommend removing the sanity check on the `unplugs[token]` from the `homogenizeMany` function.

Alleviation Acknowledged

The team acknowledged this finding.

2. Missing Emitting Events

Informational Security Analyzer

Functions update key state variables, like `allocate()`, `deallocate()`, are recommended to emit events to log the update.

File(s) Affected

Ins20.sol #37-52

```
37      function allocate(address to, uint256 amount) public onlyRegistry {
38          require(to != address(0), "Registry address cannot be address(0)");
39          unchecked {
40              mintableBalances[to] += amount;
41          }
42      }
43
44      function deallocate(address to, uint256 amount) public onlyRegistry {
45          require(
46              mintableBalances[to] >= amount,
47              "Insufficient mintable balances"
48          );
49          unchecked {
50              mintableBalances[to] -= amount;
51          }
52      }
```

Recommendation

Recommend emitting event for updating key state variables.

Alleviation Acknowledged

The team acknowledged this finding.

3. Unused Private State Variable

Informational Security Analyzer

The private state variable `binding` is used in the constructor when deploying the `Ins721` contract, however, it is never updated or used in other functions.

File(s) Affected

Ins721.sol #58-58

```
58          binding = _binding;
```

Ins721.sol #33-33

```
33      address private binding;
```

Recommendation

Recommend removing the unused private state variable `unbinding`, or declaring it as `immutable` to save gas.

Alleviation Acknowledged

The team acknowledged this finding.

4. Checking on `registry` early

Informational Security Analyzer

There is a sanity check on the `registry` from the `transferFrom` function, which could be done in the constructor, to reduce repeated checks on `registry` from the `transferFrom` function when users transfer NFT. Note that, the `registry` will not be updated after the deployment.

File(s) Affected

Ins721.sol #57-57

```
57          registry = _registry;
```

Ins721.sol #75-75

```
        require(registry != address(0), "Registry address cannot be address(0)");
```

Recommendation

Recommend bringing the sanity check on `registry` to the constructor.

Alleviation Acknowledged

The team acknowledged this finding.

Disclaimer

This report is governed by the stipulations (including but not limited to service descriptions, confidentiality, disclaimers, and liability limitations) outlined in the Services Agreement, or as detailed in the scope of services and terms provided to you, the Customer or Company, within the context of the Agreement. The Company is permitted to use this report only as allowed under the terms of the Agreement. Without explicit written permission from MetaTrust, this report must not be shared, disclosed, referenced, or depended upon by any third parties, nor should copies be distributed to anyone other than the Company.

It is important to clarify that this report neither endorses nor disapproves any specific project or team. It should not be viewed as a reflection of the economic value or potential of any product or asset developed by teams or projects engaging MetaTrust for security evaluations. This report does not guarantee that the technology assessed is completely free of bugs, nor does it comment on the business practices, models, or legal compliance of the technology's creators.

This report is not intended to serve as investment advice or a tool for investment decisions related to any project. It represents a thorough assessment process aimed at enhancing code quality and mitigating risks inherent in cryptographic tokens and blockchain technology. Blockchain and cryptographic assets inherently carry ongoing risks. MetaTrust's role is to support companies and individuals in their security diligence and to reduce risks associated with the use of emerging and evolving technologies. However, MetaTrust does not guarantee the security or functionality of the technologies it evaluates.

MetaTrust's assessment services are contingent on various dependencies and are continuously evolving. Accessing or using these services, including reports and materials, is at your own risk, on an as-is and as-available basis. Cryptographic tokens are novel technologies with inherent technical risks and uncertainties. The assessment reports may contain inaccuracies, such as false positives or negatives, and unpredictable outcomes. The services may rely on multiple third-party layers.

All services, labels, assessment reports, work products, and other materials, or any results from their use, are provided "as is" and "as available," with all faults and defects, without any warranty. MetaTrust expressly disclaims all warranties, whether express, implied, statutory, or otherwise, including but not limited to warranties of merchantability, fitness for a particular purpose, title, non-infringement, and any warranties arising from course of dealing, usage, or trade practice. MetaTrust does not guarantee that the services, reports, or materials will meet specific requirements, be error-free, or be compatible with other software, systems, or services.

Neither MetaTrust nor its agents make any representations or warranties regarding the accuracy, reliability, or currency of any content provided through the services. MetaTrust is not liable for any content inaccuracies, personal injuries, property damages, or any loss resulting from the use of the services, reports, or materials.

Third-party materials are provided "as is," and any warranty concerning them is strictly between the Customer and the third-party owner or distributor. The services, reports, and materials are intended solely for the Customer and should not be relied upon by others or shared without MetaTrust's consent. No third party or representative thereof shall have any rights or claims against MetaTrust regarding these services, reports, or materials.

The provisions and warranties of MetaTrust in this agreement are exclusively for the Customer's benefit. No third party has any rights or claims against MetaTrust regarding these provisions or warranties. For clarity, the services, including any assessment reports or materials, should not be used as financial, tax, legal, regulatory, or other forms of advice.