SOLIDProof

Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

METAPOO

Audit

Security Assessment
28. February, 2022

For

Disclaimer

SolidProof.io reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc’...)

SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof’s position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	26. - 28. February 2022	• Layout project • Automated- /Manual-Security Testing • Summary

Network

Solana (Rust)

Website

https://app.metapoo.cash

Twitter

https://twitter.com/Metapoo_Solana

Description

Metapoo provides Yield Farming experience with New Generation #NFTs on #Solana

Project Engagement

During the Date, **Metapoo Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

Logo

Contract Link

v1.0

- https://github.com/metapoo/metapoo-program
    - Commit: 4daf4aa6d73e22733fd0ca194cc250021bffc97c

Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon aspossible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
    - i) Review of the specifications, sources, and instructions provided to SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
    - ii) Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    - iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SolidProof describe.

2. Testing and automated analysis that includes the following:
    - i) Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.

3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

Used Code from other Frameworks/Smart Contracts (direct imports)

Imported packages:

THROUGH NPM PACKAGE MANAGER AND YARN

```
   "@metaplex-foundation/mpl-token-metadata": "^0.0.2",
   "@metaplex/js": "^4.10.0",
   "@project-serum/anchor": "0.18.2",
   "@project-serum/associated-token": "^0.1.1",
   "@solana/spl-token": "^0.1.5",
   "yaml": "^1.10.2"
```

125 CRATE DEPENDENCIES

Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.

v1.0

Filename	SHA-1 Hash
lib.rs	98a5cf2a4a198adf213593f4a38215f0598c5ec9

Scope of Work/Verify Claims

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .rs).

We will verify the following claims:

1. Missing signer checks
2. Missing ownership checks
3. Missing rent exemption checks
4. Signed invocation of unverified programs
5. Solana account confusions
6. Re-initiation with cross-instance confusion
7. Arithmetic overflow/underflows
8. Numerical precision errors
9. Loss of precision in calculation
10. Incorrect calculation
11. Casting truncation
12. Exponential complexity in calculation
13. Missing freeze authority checks
14. Insufficient SPL-Token account verification
15. Over/under payment of loans
16. Overall checkup (Smart Contract Security)

Overall checkup (Smart Contract Security)

Tested	Verified
✓	✓

Legend

Attribute	Symbol
Verfified / Checked	✓
Partly Verified	⚑
Unverified / Not checked	✗
Not available	–

Modifiers and public functions

v1.0

Modifiers

impl **PoolAccount**

- allow_deposit
- allow_update
- find_master_nft
- update_state
- get_multiplier
- send_token
- close_token

impl **InitializePool**

- check_pool

Public functions

- initialize_pool
- update_pool
- close_pool
- owner_withdraw_reward
- add_master_nft
- initialize_member
- deposit
- boost
- unboost
- earn_reward
- claim
- claim_with_nft

Comments

- N/A

Please check if an OnlyOwner or similar restrictive modifier has been forgotten.

Audit Results

AUDIT PASSED

Critical issues

No critical issues

High issues

No high issues

Medium issues

No medium issues

Low issues

No low issues

Informational issues

No informational issues

Commented Code exist

There are some instances of code being commented out in the following files that should be removed:

Line	Comment
795	// constraint = user_reward_account.owner == user.key()

Recommendation

Remove the commented code, or address them properly.

Audit Comments

February 2022:

Contract was compiled on Ubuntu 18.04 x64 with actual Rust, Solana, NPM and Yarn packages.

Automated testing results:
✔ Compiler Optimization Passes
✔ Pointer Analysis
✔ Building Static Happens-Before Graph
✔ Detecting Vulnerabilities

No Vulnerabilities were found in the crate dependencies.
Not lint mistakes were found against 450 lint rules.

No unit tests were performed because no corresponding tests were supplied.

Solid Proofed

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY