QuillAudits

Audit Report
March, 2024

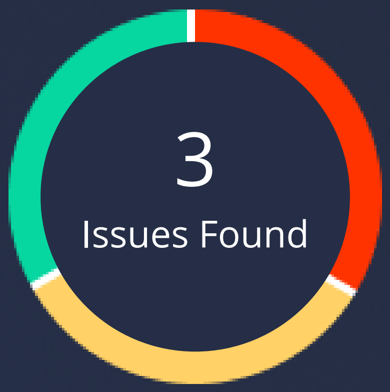
For

Table of Content

Executive Summary

Project Name	Meta Monkey
Overview	The "Elders" contract is like a special machine that creates unique digital collectibles called NFTs. It allows both the public and the owner of the machine to make these collectibles. People can buy them at a set price, and there are rules to make sure too many aren't made at once. The owner can also give them away for free to many people at once. Additionally, the owner can change some settings like the price or how many can be made.
Timeline	6th February 2024 - 15th February 2024
Updated Code Received	5th March 2024
Second Review	9th March 2024 - 11th March 2024
Method	Manual Review, Functional Testing, Automated Testing, etc. All the raised flags were manually reviewed and re-tested to identify any false positives.
Audit Scope	The scope of this audit was to analyze the Meta Monkey codebase for quality, security, and correctness.
Source Code	*https://polygonscan.com/address/0x04d70484770a93bf1bce5ef6cb33b136802b068b*
Fixed In	*https://drive.google.com/file/d/1LRfuvUJ7zIq6N3k-fejJCVGGD8s2LZBY/view?usp=share_link*

Number of Security Issues per Severity



🟥 High 🟨 Medium

🟩 Low 🟪 Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	1	0	0	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	1	1	0

Checked Vulnerabilities

- ✓ Re-entrancy
- ✓ Timestamp Dependence
- ✓ Gas Limit and Loops
- ✓ DoS with Block Gas Limit
- ✓ Transaction-Ordering Dependence
- ✓ Use of tx.origin
- ✓ Exception disorder
- ✓ Gasless send
- ✓ Balance equality
- ✓ Byte array
- ✓ Transfer forwards all gas
- ✓ ERC20 API violation
- ✓ Malicious libraries
- ✓ Compiler version not fixed
- ✓ Redundant fallback function
- ✓ Send instead of transfer
- ✓ Style guide violation
- ✓ Unchecked external call
- ✓ Unchecked math
- ✓ Unsafe type inference
- ✓ Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Hardhat, Foundry.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract’s performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

High Severity Issues

1. NFT metadata is exposed before minting

Path

https://polygonscan.com/tx/0x35b3921e08adc1a35e8e78165f4d0d2e10f4bb2da73f63273cf8f9d7f68e6b8b

Function

tokenURI

Description

This function facilitates the retrieval of the token URI containing metadata corresponding to a specific token. The URI encompasses metadata for all non-fungible tokens (NFTs), regardless of whether they have been minted. Consequently, users can peruse the metadata and discern which token ID they wish to acquire.

The metadata can be accessed via the following link structure: **https://nft.eldersgrace.io/art/metadata/1.json**. By adjusting the tokenID parameter within the JSON path—from 1 to 300—one can access the metadata for each NFT.

Users have the autonomy to prioritize the minting of NFTs based on their CARD_TYPE preferences. While the minting process unfolds sequentially from token ID 1 to 300, there may be instances where certain NFTs with common CARD_TYPEs are not minted promptly. In such cases, users may opt to front-run each other to secure NFTs with more desirable CARD_TYPEs, such as rare or epic variants.

Recommendation

It's advisable to host the metadata of the NFT only after it has been successfully minted. By leveraging IPFS (InterPlanetary File System) for hosting, users can be assured that the metadata associated with their NFTs remains immutable and resistant to alteration by the protocol.

Status

Acknowledged

Medium Severity Issues

1. Excess ether is not returned to users

Path

https://polygonscan.com/address/0x04d70484770a93bf1bce5ef6cb33b136802b068b

Function

PublicMint

Description

PublicMint takes in ether but does not return excess ether back to the sender.

Recommendation

Send back the remaining ether sent by the sender.

Status

Resolved

Low Severity Issues

1. PublicMint is not following the Check-Effects-Interaction pattern

Path

https://polygonscan.com/address/0x04d70484770a93bf1bce5ef6cb33b136802b068b

Function

PublicMint

Description

publicMintCount mapping and publicMinted variable should be updated before minting tokens, as it is calling the token receiver.

Recommendation

Call _safeMint at last.

Status

Resolved

Functional Tests

Some of the tests performed are mentioned below:

- ✓ Tested for reentrancy in PublicMint function
- ✓ Tested what if users send excess ether
- ✓ Tested how tokenURI is returned
- ✓ Tested OwnerMint and MassAirdrop function
- ✓ Tested if tokensOfOwner function reverts on bigger loops
- ✓ Tested if tokens could be transferred to different addresses

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of the Meta Monkey codebase. We performed our audit according to the procedure described above.

Some issues of High, Medium, and Low severity were found, Some suggestions and best practices are also provided in order to improve the code quality and security posture.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in Meta Monkey smart contracts. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of Meta Monkey smart contracts. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of the Meta Monkey to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem.

1000+
Audits Completed

$30B
Secured

1M
Lines of Code Audited

Follow Our Journey

Audit Report
March, 2024

For

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com