GA GUARDIAN

Cyfrin

EAS Attester & Resolver

Security Assessment

April 18th, 2025

Summary

Audit Firm Guardian

Prepared By Curiousapple, 0xCiphkey, Mark Jonathas

Client Firm Cyfrin

Final Report Date April 18, 2025

Audit Summary

Cyfrin engaged Guardian to review the security of their Cyfrin's EAS attester and custom resolver (Certifications). From the 9th of April to the 11th of April, a team of 3 auditors reviewed the source code in scope. All findings have been recorded in the following report.

For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

🔗 Blockchain network: Arbitrum Nova, Polygon, Scroll, ZKSync Era, Celo, and Blast

✅ Verify the authenticity of this report on Guardian's GitHub: https://github.com/guardianaudits

📊 Code coverage & PoC test suite: https://github.com/GuardianOrg/cyfrin-attester-team1

Table of Contents

Project Overview

Project Summary

Project Name	Cyfrin
Language	Solidity
Codebase	https://github.com/Cyfrin/cyfrin-attester
Commit(s)	Initial commit: 9b3f886973f20fe31bc34e894c5398309a81ec94 Final commit: 548eed95f5a98e5d417eccf7729730426e160253

Audit Summary

Delivery Date	April 18, 2025
Audit Methodology	Static Analysis, Manual Review, Test Suite

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
High	0	0	0	0	0	0
Medium	2	0	0	0	0	2
Low	6	0	0	4	0	2

Audit Scope & Methodology

Vulnerability Classifications

Severity	Impact: *High*	Impact: *Medium*	Impact: *Low*
Likelihood: *High*	🔴 Critical	🟠 High	🟡 Medium
Likelihood: *Medium*	🟠 High	🟡 Medium	🟢 Low
Likelihood: *Low*	🟡 Medium	🟢 Low	🟢 Low

Impact

High Significant loss of assets in the protocol, significant harm to a group of users, or a core functionality of the protocol is disrupted.

Medium A small amount of funds can be lost or ancillary functionality of the protocol is affected. The user or protocol may experience reduced or delayed receipt of intended funds.

Low Can lead to any unexpected behavior with some of the protocol's functionalities that is notable but does not meet the criteria for a higher severity.

Likelihood

High The attack is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount gained or the disruption to the protocol.

Medium An attack vector that is only possible in uncommon cases or requires a large amount of capital to exercise relative to the amount gained or the disruption to the protocol.

Low Unlikely to ever occur in production.

Audit Scope & Methodology

Methodology

Guardian is the ultimate standard for Smart Contract security. An engagement with Guardian entails the following:

- Two competing teams of Guardian security researchers performing an independent review.
- A dedicated fuzzing engineer to construct a comprehensive stateful fuzzing suite for the project.
- An engagement lead security researcher coordinating the 2 teams, performing their own analysis, relaying findings to the client, and orchestrating the testing/verification efforts.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross-referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts. Comprehensive written tests as a part of a code coverage testing suite.
- Contract fuzzing for increased attack resilience.

Findings & Resolutions

ID	Title	Category	Severity	Status
M-01	Attestor Lacks Payable Function	Logical Error	Medium	Resolved
M-02	Selective Rejection Of Low-Score Certifications	Gaming	Medium	Resolved
L-01	Redundant Delegation Functions In CyfrinAttester	Superfluous Code	Low	Resolved
L-02	Pause Behavior For certificate(tokenId) And isExpired(tokenId)	Validation	Low	Acknowledged
L-03	Unused withdrawETH And withdrawERC20 In Resolver	Superfluous Code	Low	Resolved
L-04	isValidSignature Does Not Support Contract-Based Attesters	Validation	Low	Acknowledged
L-05	Future-Proofing Cyfrin's Attester Contract	Informational	Low	Acknowledged
L-06	Base URI Not Set On Deployment	Deployment	Low	Acknowledged

M-01 | Attestor Lacks Payable Function

Category	Severity	Location	Status
Logical Error	Medium	CyfrinAttester.sol	Resolved

Description

The Ethereum Attestation Service (EAS) allows users to make attestations and send ETH if the resolver is expected to be payable. If the amount of ETH sent exceeds the value set in the attestation, EAS refunds the remaining amount to the sender, as shown in the EAS contract code.

To support cases where the remaining amount is refunded to the Cyfrin Attester, Cyfrin added a `withdrawEth` method that allows the admin to withdraw accumulated ETH. However, the current Cyfrin Attester contract does not include a `receive() payable` function or a `fallback` function to accept ETH.

As a result, if a refund is attempted, the transaction would revert. While the current resolver used by Cyfrin (Certifications) is not payable, Cyfrin may support different schemas with various resolvers in the future. In such cases, the attester might receive ETH refunds if Cyfrin deploys a payable resolver for those future schemas and attestations.

If we understand correctly, this is the reason Cyfrin included the `withdrawEth` function in the attester. If this issue goes unfixed, then for payable resolvers, Cyfrin would need to deploy a new attester with the ability to receive ETH, creating multiple on-chain identities for Cyfrin—which is not desirable.

Recommendation

Consider adding a `receive() payable` function or a `fallback` function to the Cyfrin attestor to handle incoming ETH properly.

Resolution

Cyfrin Team: The issue was resolved in PR#14.

M-02 | Selective Rejection Of Low-Score Certifications

Category	Severity	Location	Status
Gaming	Medium	Certification.sol: 67-68	Resolved

Description

Cyfrin leverages an attestation-resolver pattern to record student scores onchain, with values ranging from `s_minimumScore` to `s_maximumScore`. These scores are intended to serve as a transparent and trustable metric for talent evaluation by Cyfrin and third parties.

In the current implementation, the system uses `safeMint` to issue soulbound NFTs. This invokes the `onERC721Received` hook on the recipient's contract, giving the recipient (student) a chance to inspect the incoming certificate.

A student can program this hook to conditionally revert the minting transaction if the score is below a self-imposed threshold. This enables them to:

• Accept only high-score certificates

• Reject lower-score certificates without consequences

As a result, students can selectively curate their onchain reputation, misrepresenting their actual performance. This behavior undermines the credibility and completeness of the certification system. Evaluators relying on these onchain records might be misled, assuming that a student only received high scores, when in fact, lower scores were intentionally blocked from being recorded.

Note: With the introduction of EIP-7702, even EOAs can include temporary smart contract logic during a transaction. This means any student, including those using EOAs can now curate their onchain reputation in a misleading way.

Recommendation

Consider replacing `safeMint` with a `_mint` . Since these certificates are intended to be soulbound and non-transferable, there's no need to check for receiver compatibility for handling of NFTs.

Resolution

Cyfrin Team: The issue was resolved in PR#14.

L-01 | Redundant Delegation Functions In CyfrinAttester

Category	Severity	Location	Status
Superfluous Code	Low	CyfrinAttester.sol	Resolved

Description

In the current implementation, `CyfrinAttester` includes delegation-specific functions like `attestByDelegation` , `revokeByDelegation` , `multiAttestByDelegation` and `multiRevokeByDelegation`.

However, these actions are intended to be executed through calls made to EAS, with signature validation handled via a call to `isValidSignature` on `CyfrinAttester`.

Since EAS itself handles delegated execution, and `CyfrinAttester` only needs to expose `isValidSignature`, there is no need to implement `attestByDelegation`, `revokeByDelegation`, etc. inside `CyfrinAttester`.

These delegated functions are currently guarded by `onlyRole(ATTESTER_ROLE)`. However, if a caller already has the `ATTESTER_ROLE`, they can simply call `attest()` or `revoke()` directly, making the delegation route pointless for them.

Recommendation

Consider removing delegated action functions (`attestByDelegation`, `revokeByDelegation`, etc.) from `CyfrinAttester`.

Resolution

Cyfrin Team: The issue was resolved in PR#14.

L-02 | Pause Behavior For certificate(tokenId) And isExpired(tokenId)

Category	Severity	Location	Status
Validation	Low	Certification.sol: 168-169	Acknowledged

Description

Functions like `certificate(tokenId)` and `isExpired(tokenId)` are designed for external readers — e.g., third-party apps, scoreboards, or evaluators — to query a student's score and certificate status.

However, these functions currently do not check if the resolver is paused, and continue to return data even when the contract is paused.

Recommendation

• If the intent of `Pausable` is only to stop new attestations/revocations, then the current behavior is fine.

• But if the intent is to fully disable the use of the resolver, including read access (e.g., during a vulnerability, upgrade, or dispute period), then consider adding `whenNotPaused` to reader functions as well.

Resolution

Cyfrin Team: Acknowledged.

L-03 | Unused withdrawETH And withdrawERC20 In Resolver

Category	Severity	Location	Status
Superfluous Code	Low	Certifications.sol	Resolved

Description

The `CertificationResolver` contract includes two withdrawal functions:

• `withdrawETH()`

• `withdrawERC20(address token)`

However, the contract is non-payable and not designed to receive ETH or hold `ERC20` tokens as part of its functional logic.

Recommendation

Consider removing `withdrawETH` and `withdrawERC20` from the `CertificationResolver`. If keeping these as defensive mechanisms, clarify their intention in `NatSpec` comments and ensure they are appropriately access-controlled.

Resolution

Cyfrin Team: The issue was resolved in PR#14.

L-04 | isValidSignature Does Not Support Contract-Based Attesters

Category	Severity	Location	Status
Validation	Low	CyfrinAttester.sol: 157-158	Acknowledged

Description

If an attester role is assigned to a smart contract wallet (e.g., Gnosis Safe, kernel-based modular wallet, etc.), ECDSA recovery will fail.

These contracts do not produce ECDSA-compatible signatures and instead follow the `ERC-1271` standard for contract-based signature verification.

This means:

Delegated attestations signed by contract-based attesters will be invalid, even if they are authorized attesters.

Recommendation

If only EOAs are ever meant to hold the `ATTESTER_ROLE`, no change is needed. However, if it is expected to support smart contract wallets as attesters, update `isValidSignature` to handle nested ERC-1271 validation.

Resolution

Cyfrin Team: Acknowledged.

L-05 | Future-Proofing Cyfrin's Attester Contract

Category	Severity	Location	Status
Informational	Low	Global	Acknowledged

Description

This is regarding Cyfrin's request to understand the implications of future-proofing their attestor contract. Based on our review, Cyfrin's attester includes functions for `attest`, `revoke`, `multiAttest`, and `multiRevoke`, and supports delegations via `isValidSignature`.

Once the issue reported in `M-01` is resolved, the attester will function correctly regardless of whether the resolver is payable. One potential concern is the EAS contract itself. EAS has different versions across various deployments and may continue to deploy new versions in the future.

If EAS changes the function signature of any method or introduces new features, the attester, in its current form, will not be compatible with the new EAS deployment.

If Cyfrin intends to maintain a single attester contract with a consistent identity in perpetuity, consider making the attester upgradable to adapt to future EAS changes.

Similarly, for the resolver, once an attestation is made, the resolver address is fixed. If Cyfrin wishes to modify the resolver logic in the future, consider making the resolver upgradable as well.

Recommendation

Please note that we don't necessarily recommend upgradability, as it's a design choice for the protocol. Both approaches have their merits, and we will review whichever option you select.

Resolution

Cyfrin Team: Acknowledged.

L-06 | Base URI Not Set On Deployment

Category	Severity	Location	Status
Deployment	Low	deploy_certification.ts: 32	Acknowledged

Description

The deployment script for `Certification.sol` does sets an empty string for `baseURI`. This will lead to an empty string being stored as the NFT's Base URI on deployment of the protocol, since `_setBaseURI()` is called in the constructor.

Recommendation

Change the deployment script to set the Base URI on deployment.

Resolution

Cyfrin Team: Acknowledged.

Disclaimer

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Guardian to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Guardian’s position is that each company and individual are responsible for their own due diligence and continuous security. Guardian’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by Guardian is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract’s safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

About Guardian Audits

Founded in 2022 by DeFi experts, Guardian Audits is a leading audit firm in the DeFi smart contract space. With every audit report, Guardian Audits upholds best-in-class security while achieving our mission to relentlessly secure DeFi.

To learn more, visit https://guardianaudits.com

To view our audit portfolio, visit https://github.com/guardianaudits

To book an audit, message https://t.me/guardianaudits