Security Audit Report

Prepared by: CYBRING
Prepared for: Ferra Indexer Prices

Version: 1.0 (public version)
Date: 17th Oct 2025
Commit/Hash: c8fb072cf2790650fdb735ded11293f3cc9cddf5
Auditor: Anh Bui, Duc Cuong Nguyen

Contents

1 Introduction

This document presents the results of the initial security assessment of the `Ferra` Indexer Prices service. The main responsibilities of the service include updating the price of the oracle for supported cryptocurrencies and providing historical price data for analysis.

1.1 Objective

This security audit report was initially provided by `CYBRING` on 16th October 2025. This audit was conducted to assess the robustness, reliability, and security of the code base. The goal was to identify potential vulnerabilities, assess their impact, and provide mitigation guidance.

After the initial audit report, a reassessment was conducted on 17th Oct 2025 to verify the status of the reported issues. All issues were addressed.

1.2 Disclaimer

This security audit is not produced to replace any other type of assessment and does not aim to guarantee the discovery of all security issues within the scope of the assessment. Further, economic modeling, business-logic desirability, and market comparisons were explicitly out of scope.

While this audit was conducted with due diligence and technical proficiency, it's crucial to understand that no single audit can guarantee the absolute security of a backend service. To minimize risks, `CYBRING` recommends a multi-faceted approach involving multiple independent assessments. Please note that this report is not intended to provide financial advice.

This is the public version; some operational details and PoCs were generalized or removed.

2 Scope of Audit

The audit was conducted on commit c8fb072cf2790650fdb735ded11293f3cc9cddf5 from git repository `https://github.com/Ferra-Labs/indexer-prices`. Further details regarding The `Ferra` audit scope are provided below:

- **Codebase details:**
  - Language: TypeScript
  - Runtime: Node.js
  - Initial audit's commit hash: c8fb072cf2790650fdb735ded11293f3cc9cddf5
  - Reassessment audit's commit hash: abac74268934fad8de9822fda75c060ff16cd8bc

3 Audit Summary

`CYBRING` assessed `Ferra` Indexer Prices for security, reliability, and operational hardening. The initial review found 4 issues (2 Medium / 1 Low / 1 Informational; no High or

Critical). The Medium-risk items include exposing Swagger and potential data loss. The assessment report is summarized in Table 1. Details of findings can be found in Section 5.

Severity	Count
Medium	2
Low	1
Informational	1

Table 1: Initial assessment summary

The `Ferra` team addressed all the finding items. Summary of reassessment can be found in Table 2. As of 17th Oct 2025, no known security issues remain in the reviewed scope.

Issue	Severity	Status	Remediation Evidence
5.1	Medium	Fixed	b3e38c3
5.2	Medium	Fixed	3609669
5.3	Low	Fixed	50d855e
5.4	Informational	Fixed	04692d5

Table 2: Final assessment summary

4 Methodology

4.1 Audit Items

`CYBRING` follows a structured approach to strengthen the security posture of `Ferra` backend services:

- **Pre-auditing:** Review architecture and trust boundaries, identify integrations, and collect key artifacts (OpenAPI specs, environment/configs, CI/CD manifests).
- **Code & dependency review:** Perform manual inspection of handlers and middleware for input validation, authentication/authorization, and unsafe patterns. Run static analysis, secret scanning, and supply-chain checks for dependency risks.
- **Dynamic testing:** Conduct endpoint fuzzing and negative testing based on the OpenAPI contract to detect logic flaws, auth bypasses, CORS/CSRF misconfigurations, and DoS vectors.
- **Configuration & deployment review:** Assess HTTP headers, rate-limiting, secrets management, Docker/K8s hardening, and CI/CD security guardrails.
- **Authentication & authorization testing:** Validate token/session lifecycle, claim checks, and tenant isolation for sensitive operations.
- **Resilience & availability:** Evaluate input/time complexity limits, retry mechanisms, and graceful degradation under failure.

- **Deliverables:** Provide initial report with prioritized findings, support remediation Q&A, and re-assess fixes before issuing the final report and hardening checklist.

Details of audit items are listed in Appendix A.

4.2 Risk Rating

The OWASP Risk Rating Methodology is applied to backend issues using:

- **Likelihood:** how easily the flaw can be discovered/exploited (preconditions, attack surface, required auth/role, exploit reliability).
- **Impact:** business impact on confidentiality, integrity, availability, financial loss, privacy, and compliance.

Impact				
	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Informational	Low	Medium
		Low	*Medium*	*High*
		Likelihood		

Table 3: Overall Risk Severity

4.3 Audit Categories

- **Common vulnerabilities:** Assessed against OWASP Top 10 (Web) and OWASP API Security Top 10 (2023).
- **Advanced vulnerabilities:** Targeted exploitation of business logic, multi-tenant isolation, and chained flaws (e.g., SSRF → metadata access, authn bypass → IDOR).
- **Security best practices:** Coding standards, TypeScript strictness, schema-first validation, secure plugin configuration, deployment hardening, observability, and incident readiness.

5 Detailed Findings

5.1 Exposing Swagger

- **Location**: Swagger module
- **Description:** The Swagger UI, which documents and allows interaction with the API, is exposed in the production environment. This can reveal sensitive information about the API's structure, endpoints, and data models to potential attackers.
- **Risk:** Medium (Impact: Medium, Likelihood: Medium)
- **CWE:** CWE-489: Active Debug Code

- **Mitigation**: It is recommended to disable or restrict access to the Swagger UI in production environments. Access should be limited to authorized developers or disabled entirely to prevent the exposure of sensitive API details.
- **Status [17/10/2025]:** `Ferra` team has fixed this issue.

5.2 Potential Data Loss

- **Location**: Worker module
- **Description:** The `createPriceFeeds` task lacks a retry mechanism. If this task fails, the system will use outdated data for the next 24 hours, which may lead to stale or inaccurate price data.
- **Risk:** Medium (Impact: High, Likelihood: Low)
- **CWE:** CWE-754: Improper Check or Handling of Exceptional Conditions
- **Mitigation**: Implement a retry mechanism with an exponential backoff for the `createPriceFeeds` task. This ensures that transient failures do not result in the use of stale data.
- **Status [17/10/2025]:** `Ferra` team has fixed this issue.

5.3 Outdated Third-Party Dependencies

- **Location**: Dependency list
- **Description:** An analysis of the project's dependencies identified several third-party libraries that are either outdated or contain known security vulnerabilities. Using these vulnerable components can expose the application to various risks.
  - `axios`: Declared as 1.11.0; upgrade to version 1.12.0 or newer.
- **Risk:** Low (Impact: Medium, Likelihood: Low)
- **CWE:** CWE-1104: Use of Unmaintained Third Party Components
- **Mitigation:** To mitigate this finding, all identified dependencies should be updated to their specified secure versions. Regularly updating dependencies is a critical practice for maintaining the security and stability of the application.
- **Status [17/10/2025]:** `Ferra` team has fixed this issue.

5.4 Unused Rate Limit Plugin

- **Location**: Rate limit module
- **Description:** A rate-limiting plugin is configured but not actively being used or enforced on any routes.
- **Risk:** Informational (Impact: Low, Likelihood: Low)

- **CWE**: CWE-561: Dead Code
- **Mitigation**: It's recommended to either remove the unused plugin to reduce code complexity or properly configure and enable it on the appropriate routes to enhance the system's security posture.
- **Status [17/10/2025]:** `Ferra` team has fixed this issue.

A Appendix

Area	Representative Checks (TypeScript)
Input & Data Validation	JSON schema on all routes (AJV/TypeBox/Zod); type coercion off where unsafe; max body size; strict content types; canonicalization; defense against prototype pollution and mass assignment.
Authentication	Passwordless/OIDC/JWT/session design; token storage (cookies vs. headers); rotation/revocation; session fixation; cookie flags (HttpOnly, Secure, SameSite); MFA hooks where applicable.
Authorization & Tenancy	RBAC/ABAC enforcement on every handler; resource-scoped checks; IDOR; insecure direct joins; cross-tenant data leakage; reference monitor centralization.
Crypto & Secrets	Key management, JWKS pinning/rotation, alg selection; env secret handling (no secrets in repo/logs); KMS/secret store usage; TLS config.
Transport & Headers	HSTS, CSP, X-Frame-Options/frame-ancestors, X-Content-Type-Options, cache policy; security-header middleware (Helmet-style) configuration; redirect safety.
CORS & CSRF	Origin/host validation; allowed methods/headers; credentialed requests; CSRF defenses; preflight handling.
Business Logic	Workflow abuse, replay, re-entrancy-like sequences across endpoints, order-of-operations, race conditions, money/-points/limits enforcement.
Storage & Query Safety	SQL/NoSQL injection; Prisma/TypeORM query patterns; transaction boundaries; isolation levels; migration safety; pagination/limit caps.
SSRF & Egress	URL fetchers (DNS rebinding, IP allow-list bypass, localhost/metadata access), redirect chaInspins, file://, gopher://, and %2F tricks.
File Handling	MIME sniffing, extension checks, upload size/number limits, image processing safety, decompression bombs, antivirus hooks (if applicable).
DoS	Performance Rate limits (middleware or gateway); per-IP/account quotas; slowloris protection; timeouts; ReDoS; event-loop blocking hotspots.
Error Handling & Logging	No stack traces to clients; safe error codes; PII/secret redaction; correlation IDs; security/audit logs with tamper resistance.
Dependency & Build	SCA results; lockfile hygiene; banned/postinstall scripts; reproducible builds (`npm ci`); TS compiler strictness; `ts-node` not in prod.
Config & Deploy	Docker/K8s least privilege (non-root, RO FS); health/readiness; env var validation; feature flags; production toggles; debug endpoints disabled.
Observability & IR	Metrics/alerts for auth failures, 4xx/5xx spikes, rate-limit hits; trace sampling; runbooks and incident hooks.
Privacy & Compliance	Data minimization; retention/TTL; access logging for sensitive records; export/delete workflows; consent and notice.
Documentation & API Contract	OpenAPI accuracy; response schemas; deprecation policy; versioning; idempotency keys for mutating endpoints.

Table 4: Backend (TypeScript) audit items assessed by `CYBRING`.