Abundera.ai Security Whitepaper

Version 2.1 — February 2026

Classification: Public

Table of Contents

1. Executive Summary

Abundera.ai is a financial technology platform built to handle sensitive personal and financial data with the rigor that enterprise clients and regulated industries demand. Rather than relying on traditional centralized infrastructure, Abundera operates entirely on Cloudflare’s edge network, eliminating origin servers as a class of attack surface. Every request is processed at the edge location nearest to the user, with no single point of failure and no traditional server fleet to compromise.

The platform is protected by 908 automated attack simulations across 44 categories, 7,168 static audit checks enforced on every deployment, 1,253 unit and integration tests, and weekly OWASP ZAP DAST scans. A CREST-accredited vendor has been selected for an external human-led penetration test (expected Q2 2026). The security posture is aligned with 10 regulatory and industry frameworks including GLBA, GDPR, CCPA/CPRA, HIPAA, PCI DSS, DORA, and SOC 2.

This whitepaper describes the security architecture, controls, and compliance posture that protect Abundera’s users and their data. It is intended for security teams, compliance officers, and technical decision-makers evaluating the platform for enterprise deployment.

2. Platform Architecture

Abundera runs on Cloudflare’s serverless platform, composed of five core primitives: Pages for static site delivery, Workers Functions for compute, D1 (distributed SQLite) for relational data, KV for low-latency key-value storage, and R2 for encrypted object storage. All compute executes at the edge — there are no origin servers, no VMs, and no containers to patch or maintain.

This architecture provides several inherent security advantages. The attack surface is reduced to Cloudflare’s hardened runtime environment. There is no SSH access, no operating system to exploit, and no persistent process memory between requests. Each function invocation runs in an isolated V8 sandbox, preventing cross-request data leakage. Cloudflare’s global Anycast network provides built-in DDoS mitigation at layers 3, 4, and 7 without additional configuration.

---

3. Authentication and Identity

Abundera has adopted a passwordless-first authentication model. For direct accounts, all users authenticate via WebAuthn/FIDO2 passkeys, substantially reducing credential stuffing, password spraying, and phishing risk. There are no passwords in the system — nothing stored in a hash that could be cracked offline. Enterprise customers may alternatively authenticate via corporate SSO (OIDC or SAML 2.0), which delegates authentication to the organization’s identity provider.

Passkey registration follows a five-step verified flow: register-start, register-verify, TOTP setup, TOTP verification, and register-complete. Every account is protected by a mandatory second factor. Users enroll a TOTP authenticator during registration, and the platform generates one-time backup codes for account recovery. TOTP verification includes brute-force protection: after five failed attempts within a fifteen-minute window, the account enters a timed lockout enforced via KV, preventing automated guessing attacks.

Users can view and manage their registered passkeys from the account settings page. Each passkey displays the device name, registration date, and last used date. Users can selectively revoke individual passkeys (with re-authentication required), as long as at least one passkey remains registered. Vault and Enterprise tier accounts enforce hardware-bound passkey requirements — synced (cloud-backed) passkeys are rejected during registration, ensuring that cryptographic keys never leave the hardware authenticator.

Sensitive operations — including TOTP changes, passkey revocation, account deletion, and data export — require re-authentication. The user must present their passkey again before these actions are permitted, ensuring that a stolen session token alone cannot be used to escalate access.

Account recovery uses a multi-channel verification flow with configurable admin approval requirements. Recovery requests are time-delayed and logged to the audit trail, preventing social engineering attacks from bypassing authentication controls.

Enterprise customers can configure Single Sign-On via OIDC or SAML 2.0. Admins map email domains to identity providers (Okta, Azure AD/Entra ID, Google Workspace, OneLogin) through the admin dashboard. The OIDC flow uses standard authorization code exchange with CSRF state tokens and nonce replay protection. The SAML 2.0 flow uses HTTP-Redirect binding for AuthnRequests and HTTP-POST binding for Responses, with full XML digital signature verification (RSA-SHA256), enveloped-signature transform, exclusive XML canonicalization, condition validation with clock skew tolerance, and defenses against XML Signature Wrapping (XSW) attacks — including URI-based element verification that prevents attackers from injecting forged assertions alongside legitimate signed content. Certificates are admin-configured and never trusted from the SAML response’s KeyInfo element. User accounts are automatically linked on first SSO login by email match.

In addition to session-based and SSO authentication, the platform supports three machine-to-machine authentication methods for API integration: API key authentication (with per-key IP allowlists, tier-based rate limits, and automatic 90-day inactivity rejection), mutual TLS (mTLS) client certificate authentication (with SHA-256 fingerprint verification and per-certificate IP restrictions), and SSH public key authentication (with Ed25519/RSA key support and per-key IP restrictions). All three methods support the same authorization model as session-based access.

---

4. Session Management

Authenticated sessions use JSON Web Tokens signed with EdDSA (Ed25519, RFC 8032/RFC 8037) and short-lived access tokens (15-minute expiry) and longer-lived refresh tokens (14-day expiry). Each JWT includes a unique `jti` (JWT ID) claim, a 32-character cryptographically random hex string generated via the Web Crypto API, and a `tv` (token version) claim for KV-independent revocation. The public keys are published at `/v1/auth/jwks` in standard JWKS format, serving all active signing keys for zero-downtime key rotation.

JWT key rotation is automated on a 90-day schedule. The shard-provisioner cron generates a new Ed25519 key pair, encrypts the private key with a system-level HKDF-derived key, and stores it in D1. The old signing key transitions to `active` status (still used for verification) for a 14-day grace period before being retired. The JWKS endpoint serves all signing + active keys simultaneously, ensuring tokens signed with the previous key remain valid through the transition. Manual rotation is available via an owner-only admin endpoint for emergency scenarios.

Session revocation operates on a dual-mechanism model. The primary mechanism is KV-based: when a user logs out, the platform writes a `revoked:{jti}` entry to Cloudflare KV with a TTL equal to the token's remaining lifetime. The secondary mechanism is D1-based token versioning: each user has a `token_version` counter. When "logout all" is triggered, the counter is incremented, immediately invalidating all tokens issued with the previous version — even if KV is unavailable. The authentication middleware checks both mechanisms on every request.

Refresh token family detection provides defense against token theft and replay attacks. Each login creates a new token family, and refresh tokens are rotated on every use — each rotation invalidates the previous token and issues a replacement. If a revoked token is replayed (indicating the attacker and legitimate user both hold the same token chain), the platform revokes the entire token family and emits a critical security event. A configurable grace period (default: 30 seconds) accommodates multi-tab browser scenarios where a stale cookie may be sent concurrently, distinguishing benign multi-tab refresh from genuine token reuse attacks.

The platform supports "logout all sessions" functionality, which both revokes all outstanding JTIs via KV and bumps the user's token version in D1, providing durable session invalidation independent of KV availability. Auto-refresh logic on the client intercepts 401 responses, transparently requests a new access token via the refresh endpoint, and retries the original request, maintaining seamless user sessions without exposing long-lived credentials.

---

5. API Security

The API is protected by multiple layers of defense. Cloudflare API Shield enforces request validation against a published OpenAPI 3.0.3 schema with the default mitigation action set to `block`, rejecting malformed or unexpected requests before they reach application code. A 1MB body size limit in the root middleware guards against payload-based denial of service, returning HTTP 413 for oversized requests.

Content-Type enforcement ensures that write operations (POST, PUT, PATCH) only accept `application/json`, with explicit exceptions for multipart file upload endpoints. Requests with unexpected content types receive HTTP 415 responses. The `Vary: Authorization` header is set on all API responses, preventing CDN cache poisoning attacks where an authenticated response could be served to an unauthenticated user. ETag headers are suppressed on sensitive prefixes — including `/auth/`, `/api/financial/`, `/billing/`, and `/plaid/` — to prevent information leakage through conditional request probing.

Login and registration endpoints are protected by Cloudflare Turnstile, a privacy-preserving CAPTCHA alternative that blocks automated credential attacks. Stripe webhook processing includes KV-based idempotency checks (`stripe_event:{eventId}` with 24-hour TTL), preventing duplicate event processing and replay attacks.

API versioning is enforced via a `/v1/` prefix, with the versioning middleware forwarding request correlation IDs and preserving the original client IP through internal headers.

---

6. Security Headers

All responses include a comprehensive set of security headers, applied via the root middleware for API routes and the `_headers` file for static assets.

API responses receive a strict Content Security Policy (`default-src 'none'` with explicit `'none'` for every directive), preventing any content loading from API responses. Static pages use a permissive but audited

CSP that allowlists specific script and frame sources for Cloudflare Insights, Turnstile, Plaid Link, and Stripe.js.

The full header set includes:

- **Strict-Transport-Security** — HSTS with a one-year max-age, includeSubDomains, and preload
- **X-Frame-Options** — DENY, preventing clickjacking
- **X-Content-Type-Options** — nosniff, preventing MIME-type confusion
- **Referrer-Policy** — strict-origin-when-cross-origin
- **Permissions-Policy** — disables camera, microphone, geolocation, payment, bluetooth, USB, and 15 other browser APIs
- **Cross-Origin-Opener-Policy** — same-origin, providing Spectre-class side-channel protection
- **Cross-Origin-Resource-Policy** — same-origin, preventing cross-origin resource leaks
- **Cross-Origin-Embedder-Policy** — `require-corp` on API routes, `credentialless` on static pages
- **X-Permitted-Cross-Domain-Policies** — none
- **X-DNS-Prefetch-Control** — off, preventing DNS prefetch information leakage
- **Access-Control-Allow-Origin** — explicit origin whitelist (no wildcard `*`)

---

7. Data Protection and Isolation

Abundera separates its data architecture into two planes. The control plane, hosted in a single D1 database, handles authentication, billing, and shard management. The data plane distributes financial data across isolated, per-tier D1 shards. This separation ensures that a compromise of one tenant's data shard cannot expose authentication credentials or billing information.

Shard density scales with plan sensitivity: Core and Plus tiers share shards with up to 250 users, Pro tiers are limited to 50 users per shard, and Vault tier customers receive a dedicated 1:1 shard with no co-tenancy. Schema migrations are managed automatically by a shard provisioner worker that runs hourly, applying versioned migrations idempotently across all active shards.

All data at rest is encrypted via Cloudflare-managed encryption across D1, KV, and R2. Sensitive fields (financial data, health data, authentication secrets) are additionally encrypted at the application layer using AES-GCM-256 via the Web Crypto API with 12-byte random IVs (NIST SP 800-38D Section 8.2), explicit 128-bit authentication tags (Section 5.2.1.2), and Authenticated Additional Data (AAD) that binds each ciphertext to its user and resource context — preventing cross-context ciphertext relocation. Ciphertext is versioned for forward compatibility: text fields use a `v1:` prefix followed by base64-encoded IV, ciphertext, and tag; documents use a 3-byte binary header (magic bytes + version) followed by raw encrypted bytes.

Encryption uses a split-key derivation model: two independent secrets (`SESSION_KEY` and `ENCRYPTION_SPLIT_KEY`) are combined via HKDF (RFC 5869) — `SESSION_KEY` as IKM and `ENCRYPTION_SPLIT_KEY` as salt — ensuring that compromise of either key alone is insufficient to derive user keys. For deployments with AWS KMS configured, a cross-provider envelope encryption layer adds defense-in-depth: the local combined key is further combined with a KMS-decrypted Data Encryption Key via a second HKDF derivation, so that compromising Cloudflare infrastructure alone is insufficient to derive encryption keys. The KMS integration uses SigV4-authenticated API calls with encryption context binding (service, purpose, environment) and a 5-minute decrypted key cache with fail-hard semantics — if KMS is configured but unavailable, encryption operations fail rather than silently degrading to local-only keys.

Per-user Data Encryption Keys are derived using HKDF-SHA256 with the user ID as context — keys are derived at runtime and never stored, and compromise of one user's derived key does not affect other users. Key generation, rotation, and revocation follow a formal Key Ceremony protocol with dual-approval requirements (see `docs/KEY_CEREMONY.md`). Data in transit is protected by TLS 1.3, terminated at Cloudflare's edge.

Document uploads to R2 are encrypted at the application layer using the same AES-GCM/HKDF scheme as financial data fields — each document is encrypted with a per-user derived key before being written to R2, with the original content type preserved in R2 custom metadata. Documents are constrained to nine approved MIME types with a 2MB size limit, and stored under user-scoped key paths (`docs/{userId}/{uuid}/{filename}`) to prevent cross-user access. Filenames are validated to reject path traversal patterns and sanitized to remove dangerous characters before storage.

Every database query that accesses user data includes a `WHERE user_id = ?` clause, enforced by row-level access control in the CRUD middleware. No endpoint returns data belonging to a different user. Cross-tenant data isolation is continuously verified by automated canary tests (described in Section 10).

---

8. Rate Limiting and Abuse Prevention

Rate limiting operates at three levels: middleware-level per-IP, per-user, and per-API-key.

Middleware-level (per-IP): All write operations on API routes are rate-limited based on the client IP. Route-specific limits are applied: login and registration are restricted to 5 requests per 15 minutes, billing operations to 10 per 15 minutes, and general API operations to 30 per minute. The rate limiter fails closed — if KV or D1 is unavailable, the request is denied rather than allowed.

Per-user: Session-authenticated users are limited to 100 requests per minute, enforced per-user via KV counters. This prevents account-level abuse even if the attacker distributes requests across multiple IPs. Owner accounts are exempt to allow unrestricted platform operations.

Per-API-key: API key access uses tier-based limits. Pro-tier keys are permitted 1,000 requests per day and 10 per minute; Vault-tier keys allow 10,000 per day and 60 per minute. Multiple API keys belonging to the same user share a single rate limit bucket, preventing bypass through key rotation. Keys that have not been used for 90 days are automatically rejected, and keys without IP restrictions require an expiry date between 1 and 90 days.

Webhook receivers from external services (Stripe, Plaid, Twilio) are exempt from rate limiting — they have their own replay protection mechanisms.

---

9. Monitoring, Alerting, and Anomaly Detection

All API requests are logged to a dedicated D1 database (`abundera-logs`), separate from application data, via non-blocking `waitUntil()` calls in the root middleware. Each request is tagged with a correlation ID (`req_` prefix + 12 hex characters) for end-to-end tracing across the v1 proxy, middleware, and handler layers.

A structured security event pipeline captures 29 event types across seven categories in real time: authentication events (login success/failure, MFA failures, passkey registration/revocation, account lockouts), session events (revocation, token version mismatch, refresh token reuse), key management events (JWT rotation, revocation, encryption key rotation), administrative events (role changes, user suspension, config changes, SSO configuration), data events (export requests, deletion requests, document uploads), API events (rate limiting, schema violations, auth bypass attempts), and infrastructure events (key decryption failures, unhandled errors, KMS failures). Events are classified by severity (critical, high, medium, low, info) and written non-blocking to the `security_events` table in LOGS_DB. Critical and high-severity events are automatically escalated to all configured notification channels — email, SMS, in-app notifications, and signed webhooks — without manual intervention.

An automated behavioral anomaly detection system runs hourly as part of the shard provisioner worker, analyzing both traditional metrics and security events for patterns: brute force detection (>10 failed logins from same IP in 1 hour), potential account takeover (recovery followed by session revocation), impossible travel detection (logins from geographically distant locations within implausible timeframes), new device detection (logins from previously unseen user agents), login time anomaly detection (authentication outside established user patterns), and threshold-based checks for error rates, response times, and geographic anomalies. Alerts are delivered via Resend to security@abundera.ai, with KV-based deduplication limiting each check to one alert per hour.

All administrative actions — including user management, role changes, billing operations, and security report uploads — are recorded in a tamper-evident audit trail. Audit log entries in the `auth_audit` table are hash-chained using SHA-256, where each entry includes a hash of the previous entry. The latest hash is cached in KV for O(1) integrity verification, enabling detection of any tampering or deletion of historical records. The hash chain is anchored hourly to two independent platforms: R2 object storage (protected by Bucket Lock with 3-year retention — objects cannot be deleted or overwritten) and a dedicated GitHub repository (`abundera/audit-anchors`). Each anchor contains a segment hash covering all entries since the previous anchor, forming a secondary chain that can be verified independently of the database. Dual-platform anchoring ensures that even an attacker with full access to one storage provider cannot silently recompute the chain without detection by the other.

A public compliance status endpoint and security testing certificate page display real-time results from automated compliance checks that run hourly, covering encryption configuration, authentication enforcement, data isolation, header compliance, and audit chain integrity. Results are cached in KV and served without authentication, providing transparent verification of the platform's security posture.

Platform availability is monitored every 60 seconds from multiple global locations. Response times, uptime, and incident history are tracked continuously.

---

10. Security Testing Program

Abundera maintains a continuous security testing program with five tiers of coverage:

Pre-deploy audit (7,168 checks, 23 categories): An automated audit suite runs on every deployment, scanning all 314+ files across function handlers, HTML pages, JavaScript, and documentation. Categories include data isolation, secret leakage, authentication enforcement, security headers, XSS and injection, encryption, input validation, audit log coverage, sensitive data in URLs, open redirect protection, error disclosure, structural integrity, messaging consistency, logging hygiene, compliance consent, client trust boundaries, security tooling, dependency and supply chain security, API inventory validation, file upload security, PII protection, request bounds, and deployment configuration. A single check failure blocks the deployment.

Automated attack simulations (908 tests, 44 categories): A dedicated testing framework runs 908 automated attack simulations against the production API surface. These are scripted tests, not human-led penetration testing. Categories include JWT manipulation (algorithm confusion, x5u/jku/kid injection, signature stripping), authentication bypass, IDOR, SQL injection, XSS, command injection, SSRF (IPv6, IP encoding, cloud metadata endpoints), HTTP request smuggling, race conditions, parameter pollution, header manipulation, open redirect, business logic abuse, mass assignment, BOPLA, rate limit bypass (XFF spoofing, case variation, path encoding), resource exhaustion (ReDoS, upload bombs, sort injection), OpenAPI fuzzing, API inventory validation, billing security, webhook verification, owner authorization, token revocation, and end-to-end lifecycle flows. The suite maintains a 100% pass rate.

Unit and integration testing (1,253 tests, 33 files): Vitest runs 1,253 tests covering all middleware layers, authentication flows, billing logic, CRUD operations, rate limiting, shard management, cryptographic utilities, and webhook processing.

DAST scanning (OWASP ZAP): Weekly baseline scans run against the production site using OWASP ZAP's passive scanner via Docker. Results are automatically normalized and uploaded to the admin security dashboard. The scan covers security headers, CSP validation, CORS configuration, information disclosure, and application error detection.

Cross-tenant canary tests: Automated hourly tests verify data isolation between tenants. Two synthetic canary users (with deterministic UUIDs) are provisioned on active shards, each with encrypted test records. The canary test verifies that User A's queries cannot return User B's data, that encrypted records can only be decrypted with the correct user's derived key, and that cross-shard access is impossible. A dead man's switch monitors test execution — if canary tests fail to run for a configurable period, the system escalates via all notification channels.

A CREST-accredited vendor has been selected for an external human-led penetration test (expected Q2 2026) to complement the internal automated program as part of the SOC 2 readiness roadmap.

---

11. CI/CD Security Pipeline

Every code change passes through ten automated security jobs before reaching production:

1. **Test suite** — 1,253 vitest unit and integration tests, preceded by a cryptographic constants sync verification that ensures all worker copies match the canonical source
2. **ESLint** — Static analysis with the eslint-plugin-security ruleset
3. **Security audit** — `npm audit` for known vulnerability detection
4. **Semgrep** — Static application security testing (SAST) with custom rules for D1 isolation, auth coverage, PII in logs, missing AAD on encryption, SELECT * on sensitive tables, and timing-unsafe comparisons
5. **Gitleaks** — Secret detection across the entire commit history
6. **SBOM generation** — Software Bill of Materials (CycloneDX) for supply chain transparency, with automated vulnerability scanning at high severity threshold
7. **CodeQL** — GitHub's semantic code analysis for vulnerability patterns
8. **OpenSSF Scorecard** — Supply chain security posture scoring
9. **Socket** — Dependency risk analysis for compromised or typosquatted packages
10. **SLSA Provenance** — Build attestation via `actions/attest-build-provenance`, generating signed provenance records for every production build on main

The pre-deploy audit script (7,168 checks) serves as the final gate — even if all CI jobs pass, a failing audit check blocks deployment.

---

12. Supply Chain Security

Abundera uses only four production dependencies: a WebAuthn server library, an AWS SigV4 signing client (for KMS integration), a QR code generator (for TOTP enrollment), and the Stripe SDK. All four are pinned to exact versions with no semver ranges, ensuring reproducible builds and preventing supply chain attacks through malicious minor or patch updates.

Development dependencies are limited to three: ESLint, eslint-plugin-security, and Vitest. The total dependency surface is deliberately minimal — reducing the attack surface to seven total packages.

Dependency scanning runs daily via GitHub Dependabot with automated pull requests for security updates. The CI pipeline includes Socket for dependency risk analysis, which detects compromised packages, typosquatting, and suspicious maintainer changes. The OpenSSF Scorecard job evaluates overall supply chain security posture on every push. SLSA build provenance attestation runs on every merge to main, creating a signed chain of custody from source commit to deployed artifact.

---

13. Compliance Framework

Abundera maintains alignment with 10 regulatory and industry frameworks:

- **MVSP 24/25** — Minimum Viable Secure Product baseline (self-assessed)
- **GLBA** — Written Information Security Program (WISP) with designated qualified individual and vendor risk assessments
- **GDPR** — Data Protection Impact Assessment (DPIA) per Article 35, Records of Processing Activities (ROPA) per Article 30
- **CCPA/CPRA** — Notice-at-collection during registration with four required consent checkboxes, data export and deletion on request
- **HIPAA-Aligned** — Administrative, physical, and technical safeguards aligned to HIPAA Security Rule, applied to health-related financial data (not a covered entity)
- **PCI DSS** — Card data never touches Abundera servers; Stripe handles all payment processing
- **E-SIGN Act** — Electronic communications consent with clear opt-in
- **DORA** — All five pillars of the EU Digital Operational Resilience Act

- **EU AI Act** — Self-assessed as not high-risk per Annex III. AI usage limited to rule-based automation (threshold-based anomaly detection, error rate alerting) — no machine learning models in production. No AI involvement in financial decision-making, investment recommendations, or creditworthiness assessments. All financial computations use deterministic algorithms. Classification to be reassessed if ML models are introduced
- **SOC 2 Aligned** — All trust service criteria controls implemented; formal audit engagement planned

The platform operates with six sub-processors: Cloudflare (infrastructure), Stripe (payments), Plaid (financial data aggregation), Resend (transactional email), Twilio (SMS), and Amazon Web Services (KMS envelope encryption only — no user data stored). A Data Processing Agreement documents all sub-processor relationships, including data categories, processing purposes, and security obligations.

---

14. Incident Response and Business Continuity

The Business Continuity Plan establishes a Recovery Point Objective (RPO) of 1 hour and a Recovery Time Objective (RTO) of 2 hours. A formal change management policy governs all production modifications, and an access review process ensures that privileges remain appropriate over time.

Incident response follows a tiered severity model with defined SLAs:

- **Critical (P1)** — Platform-wide outage or active data breach: 15-minute response, 1-hour remediation target
- **High (P2)** — Significant feature degradation or potential security event: 1-hour response, 4-hour remediation target
- **Medium (P3)** — Minor issues affecting a subset of users: 4-hour response, 24-hour remediation target
- **Low (P4)** — Cosmetic or non-impacting issues: next business day

A dead man’s switch monitors founder availability as a continuity safeguard. Login heartbeats are tracked via KV, and the shard provisioner checks hourly. If no heartbeat is detected within configurable thresholds (48-hour warning, 72-hour escalation, 7-day critical), graduated alerts are sent via all notification channels. This ensures that operational continuity plans are triggered automatically if key personnel become unavailable.

Account deletion follows a strict six-step sequence: shard data removal, shard pool record update, control plane record deletion, KV session revocation, auth_users table deletion, and admin audit logging. A `deleted:{userId}` KV key with a 24-hour TTL blocks any authentication attempts during the post-deletion window, preventing race conditions between deletion propagation and cached sessions.

---

15. Data Retention and Lifecycle

Data retention periods are defined per category in a formal Data Retention Policy:

- **Request logs** — 90 days (operational monitoring)
- **Audit logs** — 3 years (compliance and forensics)
- **Audit anchors** — 3 years (R2 Bucket Lock, immutable)
- **Financial records** — 7 years (regulatory requirement)
- **Access tokens** — 15 minutes (automatic expiry)
- **Refresh tokens** — 14 days (automatic expiry)
- **API keys** — Rejected after 90 days of inactivity
- **Rate limit counters** — TTL-based, automatic KV expiration
- **Cache entries** — 30 seconds to 600 seconds depending on endpoint sensitivity

Post-deletion authentication blocks persist for 24 hours via KV. Session revocation entries expire with the original token’s remaining lifetime. Cache invalidation is triggered explicitly by write operations and Stripe webhook events, ensuring stale data is never served after mutations.

---

16. Accepted Security Trade-offs

Security engineering involves trade-offs. The following are deliberate, documented decisions:

Subresource Integrity (SRI) on CDN scripts: Cloudflare-managed scripts (Web Analytics and Turnstile CAPTCHA) are loaded without SRI integrity attributes. These scripts are auto-updated by Cloudflare — pinning a hash would break functionality on every update cycle. Both scripts are served from Cloudflare's own infrastructure under the same trust boundary as the hosting platform.

COEP credentialless on static pages: Full `Cross-Origin-Embedder-Policy: require-corp` would require all cross-origin resources (Plaid Link, Stripe.js, Turnstile iframes) to send their own CORP response header, which they do not. Using `credentialless` allows cross-origin loading without credentials while still enabling cross-origin isolation. API routes use the stricter `require-corp`.

Worker crypto duplication: Workers (shard-provisioner, webhook-delivery) cannot import modules from the main application. Cryptographic functions and constants are duplicated in worker code with sync markers referencing the canonical source. An automated CI verification step ensures constants remain synchronized across all copies, failing the build if drift is detected.

For questions about this whitepaper or Abundera's security posture, contact security@abundera.ai.

A public security testing certificate is available at abundera.ai/security/certificate.