Security Architecture Narrative

CredX Tech Inc

September 2026

Contents

Table 1: Control satisfaction

Standard	Controls Satisfied
TSC	CC6.6, CC6.7, CC7.1, CC7.2

Table 2: Document history

Date	Comment
Sep 3 2026	Rewritten with product security architecture
Sep 4 2026	Replaced with SOC 2 narrative v1.0
Sep 4 2026	Verified exceptions; added authorization finding
Sep 4 2026	Recorded confirmed storage-layer encryption; split from application layer

1 Purpose and Scope

This Security Architecture Narrative describes the security design, controls, and **known control gaps** of the CredX Tech Inc. platform in support of CredX's SOC 2 readiness effort. It is written for the Trust Services Criteria scope of Security, Availability, and Confidentiality, and is intended for CredX leadership, prospective auditors, and enterprise partners performing security due diligence.

CredX is a pre-revenue, Calgary, Alberta-headquartered fintech operating an **Embedded Value Platform (EVP)** connecting merchant POS systems to credit and lending partners. CredX is a non-bank technology provider: regulated processors, sponsor banks, and licensed lenders handle settlement, custody, lending, underwriting, and AML/KYC obligations.

Scope: the consumer web application, the merchant mobile application, the administrative dashboard, the POS device application, digital wallet integrations with dynamic QR/barcode credentials (Apple Wallet, Google Wallet), the CredX backend hosted on Heroku Private Spaces (Canadian region), and the third-party integrations making up the CredX ecosystem.

This narrative describes controls **functionally**. It does not name source repositories, files, modules, schema fields, or configuration values — those change continuously, and naming them would both date the document and disclose exploitable detail without improving a reader's understanding of the control environment. Where a finding was established by internal code review, that is stated; the location is held in CredX's engineering records.

Certification status. CredX Tech Inc. is **not currently SOC 2 certified**. CredX is pursuing SOC 2, targeting Type I first and Type II thereafter. All statements describing controls as *specified* reflect requirements documented in CredX's Security Requirements Specification and related policy artifacts; **such controls require production verification prior to audit fieldwork unless explicitly stated as confirmed.**

2 Security Objectives

- **Confidentiality, integrity, and availability (CIA)** of all system data, across mobile apps, digital wallets, and backend services.
- **PII protection across its full lifecycle** — collection, transmission, storage, use, and deletion — consistent with CredX's data classification scheme (Public, Internal, Confidential, Restricted).
- **Prevention of QR reuse, screenshot capture, and replay** for the dynamic QR/barcode payment credential used at the point of sale.
- **Defense-in-depth** suitable for enterprise and eventual carrier-grade partners, layering controls across the mobile client, network, API, and data tiers.

- **Forward compatibility** — support for future enterprise-grade and regulated deployment scenarios without requiring fundamental re-architecture.

These objectives are **design targets** stated in the Security Requirements Specification. Sections below identify specific areas where production implementation does not yet fully meet them.

3 Threat Model

Threat	Description
QR screenshot / reuse	An attacker captures a static image of the dynamic QR/barcode credential and attempts to reuse it after the legitimate transaction window
Man-in-the-middle (MITM)	Interception or tampering with traffic between the mobile app, wallet, and backend, or between backend and POS/processor integrations
Credential theft / account takeover	Compromise of customer or merchant credentials leading to unauthorized account access
API abuse / replay	Automated or scripted abuse of CredX APIs, including replay of previously captured requests
Compromised devices (rooted/jailbroken)	Mobile devices with compromised OS integrity used to bypass client-side controls or extract secrets
Insider threats	Misuse of legitimate internal access by employees or contractors
Misconfigured cloud resources	Human error in configuring hosting, storage, or network resources leading to unintended exposure

This threat model is **specified** in CredX's Security Requirements Specification. It has **not yet been validated through a formal, documented threat-modeling exercise** (for example a STRIDE workshop) or third-party review.

Two threat categories are absent from this model and are evidenced in the codebase:

- **Broken object-level and function-level authorization** — the class of flaw described in the authorization finding below. This is OWASP API Security Top 10 A01/A05, a framework this narrative names as an alignment target.
- **Browser-context threats against the consumer web application**, which the mobile-centric model above does not cover.

4 Mobile Application Security

App hardening — binary obfuscation and anti-tampering / integrity checks; detection of rooted (Android) or jailbroken (iOS) devices; disabling of debugging

and screen-recording capability where the platform supports it.

Secure storage — no on-device storage of raw payment credentials; tokens and secrets held in iOS Keychain / Android Keystore, using platform-backed encryption for locally cached sensitive data.

Authentication and session management — PIN, biometric, or passcode authentication at the app level; sessions bound to a specific device identifier; short-lived access tokens with refresh tokens rather than long-lived static credentials; automatic session invalidation on logout, device change, or detected compromise.

Secure communication — HTTPS/TLS 1.2 or higher enforced for all network communication; certificate pinning between the mobile app and CredX backend; rejection of weak or deprecated cipher suites.

These controls are specified. Production implementation — confirmation that certificate pinning and root/jailbreak detection are active in shipped app builds — has not been independently verified and should be confirmed prior to audit fieldwork.

Scope correction. CredX's mobile application is the **merchant** application, not a consumer one; the **consumer** experience is a **web** application. The mobile hardening controls above therefore protect the merchant application only. **The consumer credential-presentation path runs through a browser**, where binary hardening, device-integrity detection, certificate pinning, and platform keystore storage do not apply. A browser-appropriate threat model — cross-site scripting, session fixation, token storage in web contexts, clickjacking — is **not currently documented** and should be added. The merchant application does use platform-backed secure storage for local secrets.

5 Dynamic QR Code Security

The in-store payment credential is a **dynamic, time-bound QR code / barcode** delivered through Apple Wallet or Google Wallet, rather than a static or reusable code. This design is central to CredX's fraud-prevention posture:

- **Time-bound and single-use** — each code is valid for a short window and intended for one transaction only.
- **No raw PII or payment data in the payload.**
- **TTL 30–60 seconds + HMAC signature + session reference** — each code embeds a short-lived token, a cryptographic HMAC signature, and a transaction/session reference ID.
- **Backend validation** — the backend validates expiration, signature, and device-wallet binding before resolving a scan to a user and active loan.
- **Immediate invalidation** — tokens are invalidated immediately after a successful scan to prevent reuse.
- **Screenshot / replay protection** — because the code rotates and expires quickly and is bound to session state validated server-side, a screenshot of

a prior code is designed to fail validation on reuse.
- **Approved wallet contexts only** — the QR/barcode is rendered only inside Apple Wallet PassKit or Google Wallet, not as a freestanding image elsewhere in the app.

Wallet integration. *Apple Wallet*: PassKit-based pass with pass certificates and APNs token-based push updates; the user identifier travels inside the scannable barcode, and **the pass face does not display balance or credit limit**. *Google Wallet*: service-account JWT-based integration using the same rotating-barcode design principle.

The TTL, HMAC signing, and single-use invalidation behavior above are specified. ***Live fraud simulation of screenshot/replay scenarios against the production system has not yet been confirmed as completed.***

6 Identity and Access Management

Control	Specification
Least privilege	Access to systems and data is scoped to the minimum required for a given role
Role-based access control (RBAC)	Permissions are assigned by role rather than granted ad hoc to individuals
Account lockout	Accounts lock out after a defined number of repeated failed authentication attempts
MFA for admin/operational accounts	Multi-factor authentication is required for administrative and operational accounts
Production access restriction	Access to production systems is restricted to authorized personnel only
Admin action auditing	All administrative actions are intended to be logged and auditable

Because the Technical function both administers access and reviews it, with no independent reviewer, formal enforcement mechanisms — automated lockout thresholds, MFA enrollment records, admin-audit-log completeness — have ***not yet been independently confirmed as fully implemented in production*** *and should be verified prior to audit fieldwork.*

7 Fraud Prevention and Risk Controls

Velocity checks on QR scans and payment attempts; device fingerprinting; anomaly detection on transaction and account behavior; geo/IP reputation analysis; transaction risk scoring; step-up verification for higher-risk actions.

These are specified requirements. ***CredX does not yet have a live monitoring/SIEM capability****, which limits real-time enforcement of anomaly detection and geo/IP reputation analysis in production today; this is addressed in Phase 2 of the roadmap.*

8 Infrastructure and Backend Security

CredX's backend is hosted on **Heroku Private Spaces, Canadian region**, selected specifically for Canadian data residency. All PII and payment-related workloads are confined to the Canadian Private Space.

- **Environment isolation** — development, staging, and production environments are separated via distinct Private Spaces or strict network segmentation.
- **Inbound traffic restriction** — inbound traffic is restricted using IP allowlists, private networking, and internal routing.
- **HTTPS-only** — all traffic is served over HTTPS using Heroku-managed SSL.

Note on hosting provider. An earlier, less-accurate marketing document referenced **AWS hosting and a Zero Trust Architecture**. The verified, current source of truth is **Heroku Private Spaces (Canadian region)**. This narrative uses that as the authoritative infrastructure fact and does not represent CredX as AWS-hosted.

Environment isolation and inbound traffic restriction are specified controls. Production configuration — current IP allowlist rules, confirmation of full network segmentation between environments — has not yet been independently verified.

9 Secrets and Configuration Management

- Secrets and configuration values are managed exclusively through **Heroku Config Vars** or other approved secret managers.
- API keys, encryption keys, and signing secrets are subject to **regular rotation**.
- Keys are **separated per environment** so a compromise in one environment does not expose another.
- Secrets are **never committed to source control**.

The actual rotation cadence and a source-control secret-scanning audit have not yet been independently confirmed as implemented in production.

10 API Security

- All CredX APIs require authentication and authorization.

- **OAuth 2.0 with short-lived JWT access tokens** is the specified authentication model for API access.
- **Request signing** is specified for sensitive operations.
- **Rate limiting, throttling, and abuse detection** are specified controls against automated abuse.
- **Replay protection** is specified via nonces, timestamps, and token binding.

This is distinct from certain external integration auth models: **Clover** connects via OAuth v2 with automatic refresh, while **Square** currently connects via a static, admin-pasted token with no automated refresh, expiry, or revocation detection — a tracked exception.

Independent verification of production enforcement — load-testing rate limits, confirming nonce-based replay protection is active on all sensitive endpoints — has not yet been completed.

11 Data Protection and Encryption

CredX's encryption posture must be read at **two distinct layers**, because they are in different states and conflating them would misrepresent the control environment.

Storage-layer encryption at rest — confirmed. Data at rest is encrypted at the platform layer across both stores:

Store	Contents	At-rest encryption
PostgreSQL (Heroku Private Spaces, Canadian region)	System of record — all order, payment, loan, and PII data	Platform-managed AES-256
AWS S3 (`ca-central-1`)	Merchant logo images only — no PII or transaction data	Server-side encryption, AES-256

Encryption in transit — confirmed. TLS 1.2 or higher is enforced for all network communication, with certificate pinning specified between the mobile client and the backend.

Application-layer encryption — not implemented. The Security Requirements Specification additionally requires **field-level encryption for high-risk PII attributes** and tokenization or pseudonymization of identifiers where feasible. **No application-level encryption exists** — no encryption primitive of any kind is present in the backend codebase. Every value is written to the database in the clear and relies entirely on the platform's disk-level encryption.

Database access controls: strict access controls are specified for direct database access.

11.0.1 Why the distinction matters

Storage-layer encryption at rest protects against one threat: **physical compromise of the underlying media**. It does not protect against a database read, a compromised application credential, an over-permissive API path, or an insider with query access — in all of those cases the platform decrypts the data transparently and the reader sees plaintext.

This is precisely why the two exceptions below remain open despite at-rest encryption being confirmed. A POS token stored as a plaintext column inside an encrypted database is still a plaintext token to anyone who can query the table.

11.1 Known Exception — Unencrypted Credential Storage

Stored POS integration credentials and the wallet-pass rotating-code secret are held unencrypted in the database.

Established by internal code review. The credentials CredX stores in order to call a merchant's POS provider on their behalf — the access and refresh tokens obtained during POS connection — and the secret underpinning the wallet pass's rotating barcode are written to the database in the clear, with no application-level encryption applied. **No encryption primitive of any kind is present in the backend codebase.**

By contrast, **merchant API keys are properly stored as salted hashes**, as are user passwords. The gap is specific to POS integration credentials and the wallet-pass secret, not a platform-wide absence of practice — which is what makes it a defect rather than a design posture.

Note that this is **not resolved by the confirmed storage-layer encryption** described above: these are unencrypted values inside an encrypted database, readable by anything with query access.

Priority: top-priority open remediation item for SOC 2 readiness. *Owner: Technical function. Target date: [TBD].*

11.2 Known Exception — Stored POS Credential Is Readable Through the API

This finding materially escalates the exception above, and appears in no prior CredX compliance document.

The single-merchant lookup enforces authentication but not authorization, and returns the complete merchant record — including the stored POS credential.

The conditions that combine to produce it:

- The stored POS credential is **exposed as a field on the merchant type** in the platform's API schema, rather than being withheld from the API

surface.
- The single-merchant query requires **only a valid session**. It carries **no role restriction**, unlike every sibling merchant operation, all of which are restricted to administrators.
- Session validation confirms a valid, unrevoked session for a non-blocked user. It performs **no role check**.
- The underlying lookup applies **no field selection**, returning every stored attribute of the record.

Practical impact. Any authenticated user of **any** role — including an ordinary consumer — who knows or can guess a merchant identifier can retrieve that merchant's live POS access credential in plaintext. The earlier framing that "database access is effectively equivalent to POS access" understates this: **API access is equivalent to POS access**, and the API is reachable by every logged-in consumer.

That the platform already defines a restricted public merchant type for unauthenticated use shows the field-restriction pattern is understood; it simply was not applied to this authenticated path.

Recommended remediation, in order of speed:

1. Remove the stored POS credential from the API schema entirely — no administrative function requires reading a POS token back.
2. Apply the same administrator-only role restriction that every other merchant operation carries.
3. Apply explicit field selection on the lookup, so newly added sensitive attributes are not exposed by default.
4. Then encrypt the credentials at rest, per the exception above.

Owner: Technical function. Severity: ***Critical****. Target date: [TBD].*

12 Logging, Monitoring and Auditing

Specified logging and auditing controls cover **access to PII, authentication events, QR code validations, and payment attempts**. Sensitive log fields are specified to be masked or redacted, and logs retained per regulatory and contractual requirements, including a **24-month breach record log** regardless of breach-notification outcome.

12.1 Known Exception — Audit-Log Coverage and Phone-Number Logging

Gap 1 — Integration events missing from the audit log. Audit-log writes are made for authentication, payment, credential-validation, and refund events only. The POS, underwriting-partner, SMS, and notification integrations write **no audit-log entries at all** — token refresh failures, rejected webhooks, and

failed provider calls reach only the application logger, which carries a shorter retention period.

Gap 2 — SMS recipient numbers in application logs. The recipient phone number is written to the application log on every SMS send, and again on every send failure. A recipient phone number is personal information under PIPA and PIPEDA, and logging it is inconsistent with the masking and redaction required above.

Owner: Ray Yip, CTO / Engineering. Target date: [TBD].

12.2 Monitoring and Incident Response (Product Level)

Continuous monitoring and alerting, automated alerts for suspicious QR/API activity, and the ability to revoke tokens or disable a QR instantly are **specified** product-level capabilities.

However, **CredX does not yet have a live security monitoring or SIEM capability.** In the NIST CSF 2.0 assessment below, the **Detect function scored 0 out of 4**. Closing this gap is explicitly **Phase 2** of the Layered Security Roadmap — a planned future state, not a present-day capability.

*Real-time threat-detection tooling was named in an earlier, less-detailed policy document. Given the more recent finding that Detect scored 0 of 4, any such tooling should be treated as **[unconfirmed]** rather than asserted as live in production.*

13 Security Testing and Validation

CredX's specified security testing program includes Static Application Security Testing (SAST); Dynamic Application Security Testing (DAST); mobile application penetration testing; API security testing and fuzzing; QR replay, screenshot, and fraud simulations; and quarterly independent penetration testing stated as a target in the Data Security & Privacy Policy.

Flag: SAST, DAST, mobile pentesting, API fuzzing, and QR fraud simulation are **planned/specified, not confirmed as an active, running program**. The stated quarterly independent penetration-testing cadence should be confirmed with the CTO — **it is not yet verified as actually occurring**, and this narrative does not assert that it is.

14 Compliance and Standards Alignment

Alignment is a **design target unless otherwise noted; it does not represent certification**.

Framework / Standard	Alignment Notes
PCI DSS	Applies to payment data, addressed primarily via regulated processors/partners; CredX aligns its own handling accordingly
OWASP Mobile Top 10	Referenced in mobile app security requirements
OWASP API Security Top 10	Referenced in API security requirements
SOC 2 Trust Services Criteria	This narrative set targets Security, Availability, and Confidentiality; CredX is pursuing Type I, then Type II — **not yet certified**
ISO 27001	Aspirational alignment with security-management principles; not a certification target at this stage
PIPEDA / Alberta PIPA	Alberta PIPA is the primary, "substantially similar" jurisdiction for Alberta operations; PIPEDA applies to interprovincial/international handling. CredX targets alignment with both
Data residency / sovereignty	PII and sensitive customer data are processed and stored within Canada via Heroku Private Spaces; cross-border transfers require explicit review, documentation, and approval

15 Layered Security Roadmap

Phase	Focus	Target Outcome
Phase 1 — Baseline / MVP (current phase)	TLS everywhere, short-lived tokens, dynamic QR, encrypted storage	Foundational controls in place ahead of production launch. The plaintext-credential gap and the Detect gap are **Phase 1 open items still being closed**
Phase 2 — Enterprise Security	Enhanced monitoring and SIEM	Closes the Detect gap — introduces live security monitoring capability, moving the NIST CSF Detect score above 0/4
Phase 3 — Carrier-Grade	Hardware Security Modules (HSMs), zero-trust networking	Phased in as the platform matures, to support enterprise and carrier-grade / regulated deployment without requiring re-architecture

CredX is currently in Phase 1. Phase 2 and Phase 3 are forward-looking

roadmap commitments, not present-day capabilities.

16 NIST Cybersecurity Framework 2.0 Maturity Snapshot

CredX has performed a self-assessment against the six functions of the NIST CSF 2.0, using a 0–4 maturity scale. Full detail and methodology live in the Business Continuity Plan Narrative's Cyber Security Readiness Assessment.

Function	Score (0–4)	Notes
Govern	Low	No formal board or independent oversight body yet in place
Identify	Early stage	Asset and risk identification practices are still maturing
Protect	2	Raised from 1 (Ad Hoc) to 2 (Developing) based on documented — not yet fully production-verified — hardening, encryption, and API security requirements
Detect	**0**	**Biggest gap.** No live monitoring/SIEM capability exists. Phase 2 priority
Respond	Documented, untested	Incident response plan is documented with defined roles; **not yet tested via tabletop exercise**
Recover	Documented, untested	RTO/RPO targets are documented against the confirmed Heroku environment, and the backup policy is in force. **No restore test has been performed**, so recovery capability is designed and documented but not demonstrated

Overall maturity is approximately 1.5 out of 4 (Developing-leaning-Ad Hoc) — a reasonable and expected posture for a pre-revenue company at this stage, but **not yet production-verified and should not be represented as more mature than described here**.

17 Summary of Open Security Exceptions and Remediation Owners

Exception	Description	Owner
Stored POS credential readable through the API	The single-merchant lookup enforces authentication but not role authorization, and returns the full merchant record including its stored POS credential. **Any authenticated user of any role can retrieve a merchant's live POS access credential**	Technical function
Unencrypted POS credential and wallet-pass secret storage	POS integration credentials and the wallet-pass rotating-code secret are stored unencrypted, unlike merchant API keys, which are salted-hashed. No encryption primitive exists in the backend	Technical function
Detect / monitoring gap (NIST CSF Detect = 0 of 4)	No live security monitoring or SIEM capability. Automated alerting on suspicious credential-validation or API activity is specified but not implemented. Phase 2 of the roadmap	Technical function
Square token lifecycle gap	Square connects via a static administrator-supplied token with no automated refresh, expiry, or revocation detection, unlike Clover's OAuth model with auto-refresh and recovery	Technical function
Integration events missing from audit log	Audit-log writes cover authentication, payment, credential-validation, and refund events only; POS, underwriting-partner, SMS, and notification integration failures are not recorded	Technical function
SMS recipient numbers in application logs	Recipient phone numbers are written to the application log on every send and every failure, inconsistent with specified log masking and redaction	Technical function
Application-layer field encryption not implemented	Storage-layer at-rest encryption is confirmed, but the specified field-level encryption for high-risk PII attributes has no implementation	Technical function

Exception	Description	Owner
Object-storage encryption is implicit	Uploads rely on the storage bucket's default server-side encryption rather than requesting it explicitly. Objects are encrypted in practice, but an explicit setting or enforced bucket policy is stronger audit evidence	Technical function
Penetration-testing cadence unconfirmed	Quarterly independent penetration testing is stated as a policy target; actual occurrence has not been confirmed	Technical function — to confirm
Undocumented POS device application	A native application deployed to merchant POS hardware exists in the estate but appears in no system boundary, sub-service inventory, or security assessment	Technical function

Target dates for all items are **[TBD]**.

These exceptions, together with items tracked in the Control Environment and Business Continuity Plan Narratives — unexecuted sponsor-bank/lender contracts, no formal board oversight — form CredX's **consolidated remediation backlog** ahead of SOC 2 audit fieldwork.

Auditors and leadership should treat this table, not marketing materials, as the authoritative statement of open security exceptions.

18 References

18.1 Narratives

- System Description
- System Architecture Narrative
- Control Environment Narrative
- Organizational Narrative
- Products and Services Narrative
- Business Continuity Plan Narrative

18.2 Policies

- Encryption Policy
- Log Management Policy
- Remote Access Policy
- Security Incident Response Policy
- Vulnerability Management Policy

- Workstation Policy
- Asset Management Policy
- Security Awareness Training Policy