# DataSitr Security & Compliance Overview

**Generated:** 2026-04-21T05:00:43Z

**Source document:** docs/security-compliance-overview.md

**Git commit:** c656136

**Generator:** [operator-tooling]

**Benchmark artifact:** docs/generated/pii_benchmark_latest.json (generated 2026-04-20T03:31:24Z, Arabic NER loaded, 1K p95 31.48 ms)

---

## What DataSitr protects

DataSitr helps organizations keep personal data inside Saudi Arabia when using external AI services. It implements technical controls to support PDPL-aligned handling by detecting, anonymizing, and routing data through a three-lane privacy model.

**Core design principle:** Raw personal data is not sent to cross-border AI providers by design. External AI providers only receive detector-sanitized text after the green-lane transformation path and post-transformation rescan. The current public pilot default for that path is typed placeholders. Amber/red routing is intended to remain on operator-configured in-Kingdom paths that DataSitr does not independently verify. Residual contextual re-identification risk and legal reliance still require customer and legal review.

---

---

## The three-lane privacy model

### Green lane — anonymized, external AI

- All PII entities detected with high confidence (>=85%)
- Direct identifiers replaced with typed placeholders before eligible external processing
- A post-tokenization scan verifies no original PII text remains
- Only used for safe use cases (summarization, classification, sentiment analysis)
- The anonymized text is sent to external AI (OpenAI, Anthropic, Google)
- After receiving the AI response, original PII is restored from an encrypted in-Kingdom vault
- A transfer register entry is written per PDPL cross-border transfer requirements

### Amber lane — pseudonymized, in-Kingdom AI

- Used when detection confidence is mixed or the use case requires extra caution

- PII is replaced with typed placeholders (linkable within the session but not to real identifiers)
- Routed to operator-configured in-Kingdom AI paths
- No cross-border transfer occurs

### Red lane — raw PII, in-Kingdom AI

- Used for PDPL Article 1(11)-defined sensitive data and other highest-risk categories: health, biometric, genetic, criminal record, ethnic origin, religious belief, political opinion, plus tenant-policy escalations such as credit or location-tracking signals
- Also used when anonymization quality cannot be assured
- Raw text routed to the strictest in-Kingdom handling path available
- Data stays on Saudi-hosted infrastructure when the configured provider path is in-Kingdom
- Strictest compliance controls applied

### Block — request rejected

- Tenant policy explicitly forbids the processing
- No suitable provider available for the assigned lane
- Policy violation detected

---

## What data stays in-Kingdom

| Data type | Location | Leaves Saudi Arabia? |
|---|---|---|
| Raw PII | Encrypted vault (PostgreSQL in shared-state mode; SQLite only in local single-node mode) | Never |
| PII-to-token mappings | Vault, in-Kingdom only | Never |
| Processing records | Append-only compliance logs (JSONL in file mode, PostgreSQL in shared-state mode) | Never |
| Transfer register | Append-only compliance logs (JSONL in file mode, PostgreSQL in shared-state mode) | Never |
| API keys (hashed) | in-Kingdom only | Never |
| Anonymized text (green lane) | Sent to external AI | Yes — detector-sanitized text only; residual contextual re-identification risk still requires review |
| Pseudonymized text (amber lane) | operator-configured in-Kingdom AI path | Intended to remain in-Kingdom; provider residency is not independently verified |
| Raw text (red lane) | strict in-Kingdom AI path | Intended to remain in-Kingdom; provider residency is not independently verified |

---

## How tokenization and vaulting work

### Tokenization

1. The PII detector identifies entities in the input text (names, IDs, phone numbers, IBANs, etc.) using Presidio, spaCy NLP, Saudi-specific pattern recognizers, and the configured local Arabic semantic backend when enabled
2. Each detected entity is replaced with a typed placeholder: `[[PERSON:01]]`, `[[SA_NATIONAL_ID:02]]`, etc.
3. The original values are stored in an encrypted vault, keyed by the placeholder token and the tenant's customer ID
4. A post-tokenization scan (`check_true_anonymization`) re-examines the masked text to verify no original PII substrings remain

**Optional pseudonymization mode:** For supported Saudi structural identifiers, DataSitr also supports FF1 structural surrogates when `SV_TOKENIZATION_MODE=fpe` is explicitly enabled. That mode remains reversible pseudonymization rather than anonymization and is not the default public pilot claim for external processing.

### Vault encryption

- Algorithm: AES-256-GCM (authenticated encryption)
- Key derivation: per-tenant keys derived via HKDF-SHA256; legacy rows may still use the older SHA-256 derivation
- Legacy KDF visibility: `vault.stats()` now reports `legacy_kdf_v1_rows` and `kdf_version_counts`; operators can migrate legacy rows with `[operator-tooling]`
- Each encryption operation uses a random 12-byte nonce
- Tokens auto-expire after a configurable TTL (default: 24 hours)
- Cross-tenant retrieval returns nothing — a tenant cannot access another tenant's vault entries

### Rehydration

After the AI provider returns a response:

1. The rehydrator scans the response for placeholder patterns
2. Each placeholder is looked up in the vault (scoped to the requesting tenant)
3. Original PII values are restored in the response
4. The response is rescanned to check for any PII that may have leaked through the AI's output

---

## Subject rights support

PDPL grants data subjects rights over their personal data. DataSitr currently implements:

| Right | Status | How it works |
|---|---|---|
| **Right of access** | Implemented | `export_subject_data()` returns all processing records and vault entries related to a subject identifier for a given tenant |
| **Right to erasure** | Implemented | `delete_subject_data()` removes vault entries for a subject and logs the deletion event to `deletion_log.jsonl` |
| **Right to rectification** | Implemented | `rectify_subject_data()` updates vaulted subject values in place, keeps historical records intact, and logs the action |
| **Audit trail** | Implemented | All deletions are logged with timestamp, tenant ID, subject identifier, and operator |

Subject rights operations are tenant-scoped: a tenant can only access or delete data belonging to their own subjects.

**Not yet implemented:** Right to restrict processing, automated portability export formats.

---

## Compliance record keeping

Every processed request generates a machine-readable compliance record containing:

- Request ID and timestamp
- Tenant ID
- Purpose of processing (task type)
- Categories of personal data detected
- Route decision (green/amber/red) and reason
- Detection confidence scores
- Security measures applied (encryption, anonymization method)
- Recipients (AI provider name, if external)
- Transfer description (if cross-border)
- Policy version in effect
- Legal basis actually applied to the request

Legal-basis tracking uses strict precedence:

1. Explicit per-request override when supplied by the caller

2. Tenant policy default stored in the active tenant policy
3. `legitimate_interest` only as a documented fallback when no basis is configured

Records are stored either as append-only JSONL files in local mode or as sequenced PostgreSQL rows in shared-state mode, with 5-year retention per SDAIA guidelines. Processing records and transfer register entries can be exported as a Record of Processing Activities (RoPA).

### DPIA Generation

A machine-readable Data Protection Impact Assessment (DPIA) summary can be generated on demand at `GET /v1/admin/compliance/dpia/{tenant_id}` to support PDPL Article 22 workflows for higher-risk processing.

The generator derives its output from recorded processing activity plus the tenant's latest policy snapshot. When the source data is incomplete, the payload states that explicitly with `data_completeness` warnings instead of pretending the report is exhaustive.

> [!WARNING]

> The generated DPIA is a technical summary and structured template. It does not constitute final legal sign-off, regulator approval, or an immutable external evidence record.

---

## Audit traceability

DataSitr records the `policy_version` used for each processing event and preserves tenant policy snapshots over time.

Admins can retrieve:

- `GET /v1/admin/compliance/audit-summary/{tenant_id}` for per-version processing counts, legal-basis breakdowns, and transfer-event totals
- `GET /v1/admin/compliance/evidence-pack/{tenant_id}` for a compact JSON bundle containing the audit summary, integrity status, latest policy snapshot metadata, DPIA availability, and RoPA availability
- `GET /v1/admin/compliance/bundle/{tenant_id}` for a regulator-ready tenant-scoped JSON export that combines the audit summary, evidence pack, RoPA, DPIA, processing integrity, transfer-register integrity, governance-register coverage (TRA, breach, subject-rights), completeness metadata, and caveats in one artifact
- `GET /v1/admin/compliance/dpia/{tenant_id}` for a machine-readable DPIA derived from historical processing records and the latest policy baseline

What this proves:

- which policy versions were used for a tenant's recorded processing activity
- how many requests and transfer events occurred under each version

- which legal bases were recorded under each version

What this does not prove:

- hardware-backed WORM immutability for PostgreSQL-backed compliance logs
- signed or immutable coverage for every compliance surface; any signed handoff/package flow remains scoped to the generated artifact and does not by itself widen the underlying evidence boundary

`GET /v1/admin/compliance/verify-records` verifies the local JSONL hash chain in file mode. In `SV_DB_DSN` mode, DataSitr verifies a per-tenant PostgreSQL sequence/hash chain when every scoped processing record includes sequencing metadata (`seq_id`, `prev_hash`, `record_hash`). Legacy PostgreSQL rows that predate this metadata return `unsupported` rather than overclaiming continuity.

### Compliance bundle export

The compliance bundle endpoint is designed for buyer, auditor, and regulator handoff. It does not introduce new evidence on its own; it packages the current tenant-scoped compliance surfaces into one machine-readable artifact with:

- `completeness.status` and per-section status metadata
- `caveats` that remain visible instead of being suppressed
- the current backend mode (`local_jsonl` or `postgresql`)
- direct inclusion of the tenant's audit summary, evidence pack, RoPA, DPIA, integrity status, governance-register coverage, and legacy-row status

It still does **not** prove live shared-state reachability, runtime-topology correctness, Kubernetes HA behavior, regulator approval, or whole-register continuity for cross-tenant governance registers. Those remain separate deployment-level and regulator-level proof surfaces.

If a signed package/export workflow is enabled, the signature applies to the handoff artifact that was generated. It does not convert every underlying transfer, deletion, or access log into an external immutable sink by itself.

In PostgreSQL mode, the bundle remains truthful:

- DB-backed audit summaries remain supported
- integrity verification is `ok` only when scoped rows carry sequencing metadata and the chain verifies
- legacy pre-sequencing rows remain explicitly `unsupported`
- the bundle still does not prove an external immutable sink or regulator approval

### Legacy synchronization and remediation

Older PostgreSQL rows written before sequencing metadata was introduced remain

visible but not cryptographically recoverable. DataSitr does not attempt to

backfill or "repair" those rows, because doing so would create a false audit

trail.

Operators can retrieve a machine-readable report via

`GET /v1/admin/compliance/integrity-legacy-report`. The report is super-admin

only and summarizes, per tenant:

- `legacy_row_count`
- `sequenced_row_count`
- `first_legacy_created_at`
- `last_legacy_created_at`
- `sample_request_ids`
- `has_sequenced_rows`

This report proves visibility into unverifiable legacy rows. It does not prove

immutability, and it does not convert legacy rows into verified evidence.

---

## Tenant isolation

Tenants are isolated at every layer:

- **Cryptographic isolation:** Each tenant's vault data is encrypted with a unique derived key
- **Access control isolation:** API keys are bound to tenant IDs; all operations scoped to the authenticated tenant
- **Logging isolation:** Processing records, transfer register, and billing all tagged with tenant ID
- **Policy isolation:** Per-tenant policy configuration (external AI enabled, legal exceptions, retention)
- **Rate limit isolation:** Independent rate limit buckets per tenant

Cross-tenant access attempts are logged and return empty results (not errors, to prevent enumeration).

---

## Security controls summary

| Control | Implementation |
|---|---|
| Encryption at rest | AES-256-GCM, per-tenant keys |
| Authentication | Bearer token (SHA-256 hashed, constant-time comparison) |
| Authorization | Tenant-scoped; admin vs customer key separation |
| Rate limiting | Per-tenant API limits + nginx per-IP limits |
| Input validation | Pydantic v2 schemas, max text length enforcement |
| TLS in transit | TLS 1.2/1.3 via nginx (HSTS enabled) |

| | |
|---|---|
| Audit logging | Append-only JSONL with cryptographic SHA-256 hash chaining. PostgreSQL mode stores per-tenant sequencing metadata for verifiable continuity when present, but still lacks external WORM guarantees. |
| Request traceability | Request ID propagated through entire pipeline |
| Secret management | Environment-managed master key with scripted rotation; external KMS / HSM is not part of the current live posture |
| Service hardening | systemd `NoNewPrivileges`, `ProtectSystem=strict`, non-root user |
| PII leak prevention | Post-anonymization rescan, confidence-based routing, fail-closed |

---

## What is implemented vs. roadmap

### Implemented and tested on the current main branch

- Three-lane routing engine with per-tenant policy
- PII detection for Saudi patterns (National ID, Iqama, phone, IBAN, names)
- AES-256-GCM vault with per-tenant isolation
- Tokenization, rehydration, and true-anonymization verification
- PDPL Article 1(11) sensitive data classification
- Processing records and transfer register
- Subject data export, rectification, deletion, and consent-withdrawal enforcement
- Breach register, subject-rights SLA tracking, and transfer-risk-assessment workflows
- API authentication with per-tenant keys
- Per-tenant rate limiting
- Circuit breaker and retry logic for providers
- Request ID traceability end-to-end
- Full DPIA auto-generation
- RoPA CSV and PDF exports
- Consent-based legal basis tracking
- Regulator-facing compliance export/signing surfaces in the current codebase
- Verification counts drift as the suite grows. Use `docs/current-status-and-roadmap.md` as the canonical dated snapshot before repeating any totals in buyer-facing material.

### Not yet implemented (roadmap)

- Further Arabic semantic NER quality tuning on real customer text (the runtime now supports a configurable local backend and benchmark-gated alternatives)

- Cryptographic log signing externalization (immutable sink for the existing local hash chain, and implementation of verifiable ordering in PostgreSQL clusters)
- HSM / external key management for master key
- Master key rotation automation
- Provider-level token streaming
- Independent security audit / penetration test

---

## Regulatory context

DataSitr is designed for compliance with:

- **PDPL** (Personal Data Protection Law) — Saudi Arabia's comprehensive data protection law
- **PDPL Transfer Regulations** (August 2024) — Cross-border transfer restrictions
- **SDAIA enforcement guidelines** — Processing record requirements, 5-year retention

**Important caveat:** The legal question of whether truly anonymized data is exempt from PDPL cross-border transfer restrictions has not been tested in Saudi courts. DataSitr's safety-first approach (routing uncertain cases to in-Kingdom AI) mitigates this risk, but legal counsel should be consulted before relying on the green lane for production use.

---

## Related docs

- Customer Security One-Pager — executive summary for security review
- Security Questionnaire Answer Pack — pre-drafted answers to common vendor assessment questions
- Threat Model — detailed threat catalog and residual risk assessment
- Tenant Isolation — isolation invariants and testing requirements
- Architecture Overview — technical request flow and module structure
- Production Readiness Checklist — verifiable evidence checklist
- Disaster Recovery Summary — backup/restore posture and failure scenarios

*This document describes technical design intent and current operational posture. It does not constitute a warranty, service-level agreement, legal guarantee, or certification of regulatory compliance. DataSitr is designed to support PDPL alignment; it does not itself grant compliance. For the canonical list of safe and unsafe claims, contact gov@datasitr.com.*