ComplyCrawl — GDPR & SOC 2 Control Mapping

Date: 2026-05-25 **Audience:** Compliance officers, DPOs, and procurement evaluating ComplyCrawl for use in regulated workflows.

This document maps ComplyCrawl features to specific GDPR articles, SOC 2 Trust Service Criteria, and ISO 27001 controls. It's not legal advice; it's a defensible engineering crosswalk you can attach to a DPIA or vendor questionnaire.

GDPR Article Mapping

GDPR Article	Requirement (summary)	How ComplyCrawl satisfies it
Art. 5(1)(a) — Lawfulness, fairness, transparency	Processing must have a lawful basis	Consent gates require explicit `consent: true` in the run config before any fetch executes. No fetches happen on default config.
Art. 5(1)(b) — Purpose limitation	Data collected only for specified purposes	Each playbook (SEO regression / catalog monitor / link health) has a declared purpose recorded in the run config; the audit log includes purpose at every fetch.
Art. 5(1)(c) — Data minimization	Collect only what's necessary	Playbooks fetch only declared fields (titles, meta, link targets, prices). No full-page body persistence unless explicitly configured.
Art. 5(1)(d) — Accuracy	Data must be accurate and up-to-date	Run timestamps + content fingerprints make staleness verifiable. Re-runs replace prior snapshots in place rather than appending phantom data.
Art. 5(1)(e) — Storage limitation	Retention only as long as necessary	Configurable retention policy per vault. Default 90 days for findings, 365 for run summaries. Older artifacts auto-pruned.

GDPR Article	Requirement (summary)	How ComplyCrawl satisfies it
Art. 5(1)(f) — Integrity & confidentiality	Appropriate security, including encryption	Fernet (AES-128) encryption at rest; SHA-256 hash chain prevents undetected tampering.
Art. 5(2) — Accountability	Demonstrate compliance	Tamper-evident audit log is the artifact of accountability. Exportable as PDF for regulator-facing review.
Art. 24 — Responsibility of the controller	Implement appropriate measures	Default-deny architecture (no fetches without consent flag) is "data protection by design" per Art. 25.
Art. 25 — Data protection by design and default	Privacy by design	Robots.txt enforced *before* the request; consent gating; encryption at rest; minimum-data principle in playbooks.
Art. 30 — Records of processing activities	Maintain processing records	Every run logs: timestamp, purpose, domains touched, robots.txt evaluation result, fetched fields, retention class. Exportable.
Art. 32 — Security of processing	Risk-appropriate technical measures	Encryption (Fernet), pseudonymization-friendly (configurable field redaction), integrity (hash chains), availability (retry/throttle), restoration (versioned vault).
Art. 35 — Data Protection Impact Assessment	DPIA for high-risk processing	The exportable audit log + this mapping doc constitute the technical evidence section of a DPIA.

SOC 2 Trust Service Criteria Mapping

TSC	Criterion	ComplyCrawl alignment
CC6.1	Logical access controls	Vault key separate from run config; access requires both.
CC6.6	Logical access — transmission protection	TLS required for all outbound HTTP; rejected configs for plain HTTP unless explicit override.

TSC	Criterion	ComplyCrawl alignment
CC6.7	Logical access — protection of data in motion	Fetched payloads encrypted before vault write.
CC7.1	System operations — monitoring	Per-domain throttling + run summary CSV with anomaly markers.
CC7.2	System operations — security events	All policy denials (robots.txt blocks, consent failures, throttle violations) logged with full context to the audit chain.
CC7.3	Anomalies and security events	Hash-chain breakage detection — if any log line is altered, downstream hashes break verification and the next run flags it.
CC7.4	Incident management	Failed runs persist their last-known state and the reason for failure. Re-runnable from snapshot.
CC9.1	Risk mitigation — disruption	Idempotent playbook design; partial runs do not corrupt prior vault state.
CC9.2	Risk mitigation — vendor management	Source-available; you become the vendor of record post-purchase. No third-party SaaS dependency.

ISO 27001 Annex A Controls

Control	Description	ComplyCrawl alignment
A.5.1.1	Information security policies	Default-deny config + threat model included in repo (`THREAT_MODEL.md`).
A.8.1.1	Inventory of assets	Vault index lists all stored artifacts with hashes; serves as machine-readable asset inventory.
A.8.2.1	Classification of information	Playbook config declares data class (public-fetched / pseudonymized / sensitive).
A.10.1.1	Cryptographic controls policy	Fernet (AES-128-CBC + HMAC) for at-rest; TLS for in-transit. Algorithm choices documented.
A.12.1.2	Change management	Run config versioned alongside the vault; any config change creates a new chain checkpoint.

Control	Description	ComplyCrawl alignment
A.12.3.1	Information backup	Vault is filesystem-backed and rsyncable; backup is your existing backup tooling.
A.12.4.1	Event logging	Every fetch, denial, and policy check logged to the append-only chain.
A.12.4.2	Protection of log information	Hash chain is the protection mechanism; log tampering breaks verification.
A.18.1.4	Privacy and PII	Default config does not capture page bodies; explicit opt-in required for body retention.

What this does NOT cover

This crosswalk attests to *engineering controls only*. A complete compliance program also needs:

- Written policies and procedures (your org's responsibility)
- Personnel training (your org)
- Physical security (your hosting)
- Business continuity planning (your ops)
- Audit cadence (your governance)
- Legal review of robots.txt enforcement scope for your specific jurisdiction

ComplyCrawl is the technical layer. The rest is on you.

How to use this doc

1. Attach to vendor questionnaires verbatim — most procurement teams accept a controls crosswalk as evidence.
2. Cite specific rows in your DPIA's "technical and organizational measures" section.
3. Use as a checklist when configuring your deployment — every row references a feature you can verify by reading the source.
4. Show your auditor. The hash-chained audit log is the artifact they want to see at audit time.

If a control you need isn't in the table, open an issue on the repo with the specific GDPR article or SOC 2 criterion. Most additions are 10–30 lines of code given the existing event/policy infrastructure.