FINAL ARCHITECTURE AND RESEARCH WHITE PAPER

Building CCRNet with Knolo

A governed, privacy-first architecture for cybercrime case intelligence using Knolo Core and Knolo Agents

Reference product designation: CCRNet Intelligence Fabric

> Core thesis
> Knolo Core should serve as CCRNet's portable, deterministic knowledge and retrieval layer. Knolo Agents should serve as its typed, policy-gated execution layer. CCRNet must remain the evidence system of record, identity boundary, provenance model, human decision workflow and exchange-governance layer.

Authored by

Sam Paniagua

HIVE Technology Foundation Inc. / CCRNet / Knolo

Final publication • Version 1.0 • 4 August 2026

Document purpose

This paper translates the current CCRNet research direction and the present capabilities of Knolo Core and Knolo Agents into a buildable product architecture. It distinguishes the knowledge layer from the execution layer, identifies the services that must remain under CCRNet governance, and defines where authorization, human review, legal basis and evidentiary controls are mandatory.

> Authorized use only
> The proposed system is for lawful, authorized cybercrime research, fraud investigation, victim support and defensive intelligence. Agent graphs are constrained workflow executors—not autonomous investigators, public accusation engines, unrestricted surveillance tools or substitutes for legal and evidentiary review.

Abstract

CCRNet describes an active research program organized around private organization and case knowledge packs, deterministic cross-report search, human-reviewed connection suggestions, model research and opt-in sanitized intelligence exchange. Knolo Core provides versioned portable knowledge packs, deterministic lexical retrieval, scoped filters, optional lexical-first reranking, ClaimGraph expansion, LivePack overlays, patch streams and local runtimes. Knolo Agents adds typed execution graphs, least-authority policy packs, host-injected effects, checkpoints, human-in-the-loop suspension, ordered events and replay. Together they provide a stronger foundation for governed cybercrime workflows, while remaining separate from the evidence system of record. [1–8,20–26]

The recommended design keeps original evidence and rich case metadata in an encrypted system of record. It generates purpose-bound Knolo Core artifacts as reproducible search derivatives and binds agent runs to explicit graphs, authority policies, tools, knowledge snapshots and review gates. The host selects authorized packs before retrieval, records ordered events and evidence traces, checkpoints before suspension, and presents only evidence-backed suggestions for analyst review. Cross-organization sharing remains a distinct derivative pipeline that removes private evidence, applies handling labels and exports only approved indicators through formats such as STIX 2.1 and TAXII 2.1. [10–12,20–25]

The first release should be a single-organization investigator workspace with isolated case packs, exact and lexical cross-case discovery, citation-preserving evidence cards, typed workflow graphs, policy-gated tools, reviewable connection suggestions and complete auditability. Public intake, unrestricted dark-web collection, predictive person scoring, self-approving agents and raw cross-tenant exchange should remain outside the MVP.

Research snapshot

Source	What was reviewed	Status on 4 Aug 2026
CCRNet public site	Program purpose, research tracks, platform preview, developer architecture and planned APIs	Active R&D; preview concepts are explicitly non-operational
Knolo public site	Positioning, package ecosystem, local-first model and benchmark claims	Public product/documentation site
knolo-core repository	README, package metadata, builder, query, LivePack, patch-pack and runtime behavior	Public main branch; @knolo/core v3.5.0 in package metadata

Source	What was reviewed	Status on 4 Aug 2026
Knolo-Agents repository	README, core boundary, retrieval, policy, checkpoints, replay, TypeScript SDK and Rust/ICP runtime paths	Public main branch; @knolo/agents v0.1.3; early 0.1.x APIs may evolve before 1.0
External standards	FBI IC3, Europol IOCTA, NIST, OASIS STIX/TAXII, FIRST TLP and GDPR principles	Current public guidance and standards

Authorship and institutional status

Author: Sam Paniagua. Affiliation: HIVE Technology Foundation Inc. CCRNet is the foundation's specialized cybercrime research and public-safety program; Knolo Core and Knolo Agents are open-source infrastructure projects used in this proposed architecture.

HIVE Technology Foundation Inc. is a Florida not-for-profit corporation. As of 4 August 2026, it does not claim federal tax-exempt status or tax-deductibility. This publication is a research and architecture proposal, not a representation that CCRNet is operational or generally available.

> Publication status
> Final white paper, Version 1.0. Repository behavior and package versions should be revalidated against the exact releases selected for implementation.

Contents

1 Executive summary

CCRNet can become a credible cybercrime intelligence product by combining deterministic knowledge retrieval with a typed, policy-gated workflow control plane—without confusing either layer with evidence, authorization or human judgment.

The market and public-interest need is material. The FBI's 2025 Internet Crime Report records 1,008,597 complaints and approximately $20.9 billion in reported losses; cyber-enabled fraud accounted for 452,868 complaints and approximately $17.7 billion in losses. [9] Europol's 2026 threat assessment likewise describes an increasingly complex criminal environment shaped by encryption, proxies and AI-assisted operations. [13]

CCRNet's public direction already addresses the most important design problem: sensitive reports must remain isolated by default while the system still helps authorized investigators connect recurring indicators, tactics and infrastructure. The program describes separate organization and case packs, deterministic cross-report search, human-reviewed suggestions and an opt-in sanitized exchange. [1–4]

Knolo Core is unusually well aligned with this model. Its default retrieval path is deterministic and lexical, portable packs can run without a hosted vector database, and current APIs expose filters, retrieval evidence, graph expansion, immutable snapshots, active overlays and patch streams. Knolo Agents complements those primitives with typed graphs, least-authority packs, policy-gated effects, checkpoints, HITL and replay. [5–8,20–26] Together they support reproducible review and constrained deployments, including local or air-gapped use.

> Recommendation
> Build CCRNet as a governed case-intelligence platform that embeds Knolo Core for knowledge retrieval and Knolo Agents for constrained execution. Keep original evidence, identity, authorization, provenance, credentials and final decisions outside both layers. Never treat a retrieval score, graph edge or agent output as proof of wrongdoing.

The proposed product

Layer	Working designation	Purpose
Program	CCRNet	Public-benefit cybercrime research, safety resources and responsible intelligence collaboration.
Platform	CCRNet Intelligence Fabric	Shared architecture for evidence preservation, knowledge packs, search, review, governance and exchange.
Primary application	CCRNet Investigator Workspace	Case intake, evidence review, cross-case search, connection review and reporting.
Partner surface	CCRNet Indicator Exchange	Opt-in publication and consumption of sanitized, handling-labeled indicators.
Research surface	CCRNet Model Lab	Purpose-bound evaluation of retrieval, entity resolution and typology models on governed datasets.

What the MVP should deliver

- One-organization deployment with isolated case workspaces and role-based access.
- Immutable evidence preservation, source hashing and provenance manifests.
- Atomic case knowledge documents compiled into deterministic Knolo snapshots.
- Authorized multi-case search with citations, source traces and stable result ordering.

- Exact indicator matching plus reviewable lexical and graph-assisted connection suggestions.
- Live case updates through LivePack and reproducible snapshots or patch streams.
- Complete audit of pack creation, mount, query, review, export and deletion events.

What the MVP should not deliver

- Public allegations, suspect rankings or automated guilt determinations.
- Cross-tenant search or raw evidence sharing by default.
- Autonomous dark-web purchases, criminal engagement or unrestricted collection.
- Predictive person scoring, credit decisions or enforcement recommendations.
- Raw sensitive case data on the current public-query Internet Computer canister path.
- An LLM answer that cannot show the exact evidence, pack, document and retrieval trace behind it.

2 Why this product should exist

Cybercrime cases are not merely “documents to chat with.” They are evolving evidence collections with legal, privacy, handling and confidence constraints.

The operational problem

Fraud and cybercrime investigations frequently fragment across reports, emails, screenshots, payment records, chat exports, wallet addresses, usernames, domains, device identifiers and analyst notes. The same actor or operation may reuse only a subset of those elements, use different names, change channels or target different victim groups. A general-purpose chatbot is poorly suited to this setting when it cannot guarantee scope, preserve provenance or reproduce why a result appeared.

A useful system must answer more disciplined questions: Which authorized cases contain this wallet, domain, phone fragment or phrasing? Which evidence supports the connection? Was the source verified? Is the match exact, canonical, lexical, graph-assisted or model-derived? Who reviewed it? Can it be shared, and under what handling restriction?

The product opportunity

Need	CCRNet response	Why Knolo helps
Preserve sensitive case boundaries	Organization and case isolation with explicit query scopes	Portable packs and namespace/source filters support bounded retrieval.
Reproduce investigative search	Deterministic result paths and saved query traces	Lexical-first scoring and stable pack artifacts are evaluation-friendly.
Connect recurring patterns	Exact joins plus evidence-backed suggestion workflow	ClaimGraph and lexical expansion can surface candidate relationships.
Work offline or in restricted environments	Private VPC, workstation or air-gapped deployment	No external vector database or hosted retrieval dependency is required.
Share without exposing reports	Sanitized derivative indicator exchange	Exchange packs can be generated separately from private case packs.
Evaluate models responsibly	Purpose-bound de-identified corpora and reviewer labels	Versioned packs provide reproducible evaluation inputs.

Why deterministic retrieval matters

Determinism does not make a system correct, but it makes errors inspectable. When the same pack, query and options produce the same ranked candidates, reviewers can compare versions, measure regressions and reconstruct why an analyst saw a result. Semantic reranking can still be useful, but it should follow lexical grounding and should expose its contribution separately. Current Knolo query results can carry lexical, semantic and blended evidence values, which fits this requirement. [7]

> **Important distinction**
>
> Deterministic retrieval is a review property—not an evidentiary conclusion. A stable false match is still false. CCRNet must measure false connections, require supporting artifacts and record human decisions.

3 Current CCRNet product direction

The public CCRNet concept is already organized around a sensible sequence: report, observe, investigate, predict and exchange—under private-first handling and human review.

What the public site establishes

- CCRNet is framed as a public-benefit cybercrime research program rather than a commercial accusation database. [1]
- The research corpus may include voluntarily submitted incidents and lawfully accessible or authorized sources, with provenance, timestamps, confidence and handling restrictions. [2]
- The platform preview describes a governed path from reporting through prediction, and clearly states that the preview is non-operational. [3]
- The developer page proposes isolated organization and case packs, deterministic cross-report search, human-reviewed connection suggestions and sanitized indicator exchange. [4]
- The public design explicitly separates raw reports and evidence from exchangeable indicators. [1,4]

Recommended interpretation

CCRNet should be treated as the program and governance umbrella. The product should be a modular intelligence fabric whose first application is an investigator workspace. That framing avoids forcing every future capability—community reporting, source observation, partner exchange and model research—into one monolithic application.

Public concept	Product translation	Release stance
Report	Authorized intake, preservation and case creation	MVP
Observe	Purpose-bound source collection and observation packs	Limited pilot after intake controls
Investigate	Scoped search, evidence cards, timeline, graph and review queue	MVP core
Predict	Typology or campaign likelihood research with model cards	Deferred; research only
Exchange	Sanitized indicator preparation, approval and partner distribution	Partner pilot after governance

4 Knolo Core and Knolo Agents capability assessment

Knolo Core is a strong fit for portable retrieval and reproducible knowledge artifacts. Knolo Agents adds a reviewable execution control plane. Neither project replaces the surrounding CCRNet case platform, evidence system, identity boundary or human governance model.

Capabilities that map directly

Knolo Core capability	CCRNet use	Product interpretation
Versioned .knolo packs	Immutable case, organization, typology and exchange snapshots	Derived retrieval artifact, not original evidence.
Deterministic lexical query	Reproducible case search and evaluation	Default retrieval path.
Source and namespace filters	Document-class and case-local filtering	Useful inside a pack; authorization still happens before pack selection.
Optional semantic rerank	Improve ordering when lexical confidence is weak	Only after lexical candidate generation and inside approved boundary.
ClaimGraph	Bounded query expansion and candidate relationships	Lead generation, never a verdict or full entity-resolution system.
LivePack	Active case edits over an immutable base	Working overlay; serialize reviewed snapshots.
Patch packs	Incremental, base-bound updates	Synchronization aid, not a legal chain-of-custody mechanism.
Cortex memory	Append-only analyst/application memory	Separate from evidence and verified claims.
Node, Python and Rust runtimes	Web service, research pipeline and edge/offline deployment	Use parity tests across runtimes.
ICP canister adapter	Public reference packs or publishable integrity demonstrations	Do not place sensitive case evidence on current public-query path.

Current technical constraints that shape the product

The current Knolo Core `BuildInputDoc` shape contains only `id`, `heading`, `namespace` and `text`. Rich provenance, legal basis, confidence, handling labels, artifact hashes and review state therefore require an external metadata system or a future structured metadata extension. [6] Query results expose source/document ID, namespace and retrieval evidence, which is enough to resolve back into that external record. [7]

LivePack materializes a merged corpus and rebuilds it after each mutation. This produces coherent corpus-wide scoring but implies that high-write cases should batch changes or use asynchronous snapshotting rather than treating LivePack as a high-throughput transactional database. Live semantic updates are currently rejected; the live path is lexical and graph-only. [8]

Patch packs bind operations to a base fingerprint and deterministically sort actor/timestamp operations. This is useful for consistency and replay. It is not, by itself, cryptographic signer identity, non-repudiation, retention enforcement or evidentiary chain of custody. CCRNet should add signed manifests, trusted timestamps and an independent tamper-evident audit log. [8]

Knolo Agents as the governed execution layer

Knolo Agents is a separate Rust runtime and TypeScript SDK for validated agent graphs, typed state, least-authority policy, host-owned effects, ordered events, checkpoints, resume, human-in-the-loop review and replay. It explicitly keeps provider SDKs, credentials, storage backends and `@knolo/core` implementations outside the agent workspace. [20–26]

For CCRNet, this means "agent" should describe a constrained workflow program—not a free-form autonomous investigator. Each run should bind a graph and state schema to a narrow authority policy, an approved tool registry, authorized knowledge snapshot identifiers, execution budgets and a durable event/checkpoint store. Missing authority must deny by default, and external effects must remain host-owned.

Layer	Primary role in CCRNet	Boundary that must remain explicit
CCRNet application	Identity, organization/case scope, purpose, evidence records, review state and retention	Selects permissible packs and effects before an agent run starts.
Knolo Agents	Typed workflow graphs, state transitions, policy checks, budgets, HITL, checkpoints and ordered events	Does not own credentials, evidence storage, model providers or final case decisions.
Knolo Core	Builds and queries portable knowledge artifacts; provides LivePack, Cortex and ClaimGraph primitives	Knowledge retrieval is a host capability and must not silently expand authorization.
Authority pack	Capabilities, namespaces, tools, argument constraints and hard resource budgets	Least-authority declaration; not executable code and not a knowledge corpus.
Knowledge pack	Evidence-derived or reference content for deterministic retrieval	Derived search artifact; never the original evidence or legal chain of custody.
Host effects	Retrieval, exact index, storage, LLM, notification, export and external APIs	Implementations and credentials stay outside packs and checkpoints.

Recommended CCRNet agent graphs

Agent graph	Permitted effects	Mandatory gate
Intake and quarantine	Create intake record, hash artifact, request malware scan, write quarantine metadata	Human or policy release before normalization.
Normalization and extraction	Read released artifact, create excerpts/events/indicators, write provisional structured records	Analyst verification before material facts enter a case pack.
Search and connection	Call exact index and authorized Knolo retrieval; write candidate suggestions with evidence trace	Cannot accept a relationship, identify an offender or broaden scope.
Review assistant	Assemble evidence cards, summarize supported/contradictory evidence, prepare decision context	Qualified reviewer accepts, rejects or requests more evidence.

Agent graph	Permitted effects	Mandatory gate
Exchange preparation	Read accepted share-eligible indicators, sanitize, map schema and prepare release package	Two-person or organization-defined approval before publish.
Research evaluation	Read approved de-identified packs, run benchmark and write aggregate metrics	Separate dataset approval; no operational consequence or person scoring.

Early-release caveat
The current `@knolo/agents` package is version 0.1.3 and the repository describes the 0.1.x line as early, with APIs that may evolve before 1.0. CCRNet should pin exact versions, run compatibility fixtures and treat replay/checkpoint formats as migration-sensitive until stabilized. [20,26]

What remains a CCRNet responsibility

Required CCRNet subsystem	Why Knolo Core and Agents do not replace it
Identity and authorization	Tenant isolation, roles, attributes, purpose and case membership must be enforced before retrieval.
Encrypted evidence vault	Original artifacts, malware-safe handling, legal holds and per-case keys require a system of record.
Provenance and chain of custody	Collection history, hashes, transformations, reviewer decisions and exports need durable signed records.
Workflow and review	Knolo Agents supplies execution, suspension and replay primitives; CCRNet still owns assignments, reviewer identity, signed decisions, appeals and consequence policy.
Multi-pack orchestration	An agent may call retrieval, but the CCRNet host must authorize pack scope, mount snapshots, fan out queries and merge results without expanding access.
Exact indicator index	Canonical joins for wallets, hashes, domains and accounts are better served by a structured deterministic index.
Sanitization and exchange	Redaction, pseudonymization, approval, TLP handling and revocation are policy workflows.
LLM gateway	Model calls remain host-owned effects with approved providers, prompt-injection controls, citation enforcement, redaction and output policy.

Architecture rule
A knowledge pack must be reproducible from governed source records. An authority pack must grant only the minimum capabilities needed for a workflow. Neither pack contains credentials or replaces the original evidence, identity system, signed review record or independent audit log.

5 Proposed product and trust model

The system should be private by default, evidence-led, purpose-bound and explicit about every trust transition.

Design principles

1. Isolation before intelligence: select only authorized organizations, cases and source collections before any query or model call.
2. Evidence before inference: every claim or connection must resolve to artifacts, excerpts, timestamps and provenance.
3. Deterministic first: exact and lexical paths run before graph or semantic expansion; each contribution remains visible.
4. Human review for consequence: suggestions do not alter case facts, identify a perpetrator or leave the organization without approval.
5. Derivative sharing: exchange objects are newly generated sanitized records, never a direct view into private packs.
6. Purpose and retention binding: collection, use, training and sharing must each have an authorized purpose and expiration policy.
7. Local-first deployment: sensitive workloads can run in a private VPC, sovereign environment or offline workstation.
8. No model training by default: user and case data are excluded from training unless a separately governed research dataset is approved.

Trust zones

Zone	Contents	Primary controls
Zone A — Original evidence	Uploaded files, messages, exports, screenshots, source captures	Quarantine, encryption, immutable storage, hash, legal basis, malware controls.
Zone B — Structured case record	Entities, events, indicators, excerpts, confidence, reviewer state	Database constraints, row-level security, audit, retention and correction workflow.
Zone C — Knolo retrieval artifacts	Case/org/reference packs, LivePack overlays, patch streams	Signed manifests, pack registry, authorization-aware mount, cache controls.
Zone D — AI processing	Optional embedding, rerank, summarization and extraction	Approved models, data boundary, prompt-injection defenses, citation enforcement.
Zone E — Exchange	Approved sanitized indicators and relationships	Redaction, pseudonymization, TLP, partner agreements, revocation and monitoring.

Non-goals

- Publishing names or accusations based on unverified reports.
- Inferring protected attributes, political affiliation or personal character from case data.
- Automating arrest, prosecution, account closure, credit, employment or insurance decisions.
- Building a general population identity graph.
- Using victim reports as unrestricted model-training material.
- Replacing trained investigators, counsel, incident responders or victim-support professionals.

6 Reference architecture

The architecture separates the evidence system of record, the Knolo Core knowledge layer and the Knolo Agents execution layer. CCRNet authorization selects scope before a run; the agent runtime

checks policy before effects; the host owns tools and credentials; and humans retain consequential decisions.

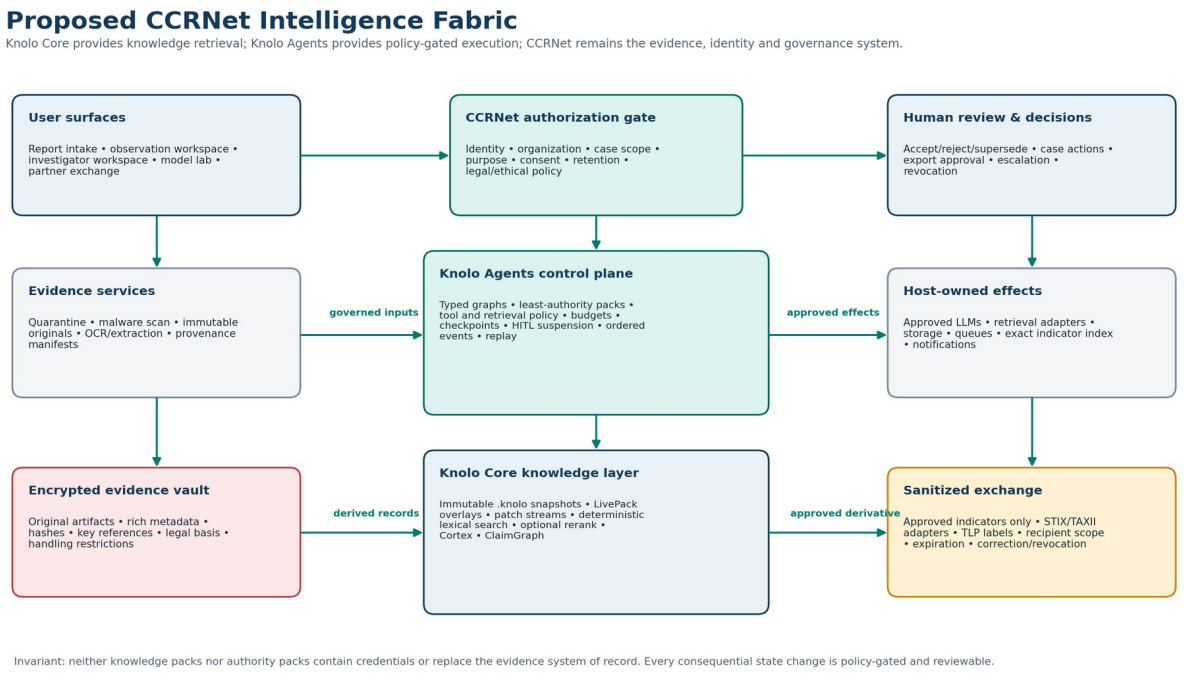

Figure 1. Proposed CCRNet architecture integrating Knolo Core and Knolo Agents while preserving separate evidence, identity, review and exchange boundaries.

Core services

Service	Responsibility	Suggested implementation
Identity & policy	Organizations, roles, case membership, attributes, purpose and consent	OIDC/SAML, policy engine, PostgreSQL row-level controls.
Intake gateway	Uploads, forms, API submissions and source imports	Next.js/TypeScript API with signed upload URLs.
Evidence vault	Immutable originals, encryption, hashes, manifests and legal holds	S3-compatible object storage + KMS/HSM + WORM option.
Normalization workers	OCR, parsing, entity/indicator extraction, canonicalization and redaction	Python workers in isolated queues; deterministic transforms where possible.
Case metadata store	Cases, artifacts, entities, events, indicators, reviews and retention	PostgreSQL with append-only audit outbox.
Pack builder & registry	Build, sign, version, store and activate Knolo snapshots or patches	Node service using @knolo/core; object store + registry DB.
Search orchestrator	Authorize pack scope, mount/cache, query, merge and trace	TypeScript/Node; exact index + Knolo adapters.

Service	Responsibility	Suggested implementation
Agent runtime & policy	Compile and execute typed graphs; validate authority; enforce tool/retrieval budgets; checkpoint, suspend, resume and emit ordered events	@knolo/agents for graph builders and clients; native Rust runtime for authoritative scheduling; durable event/checkpoint store.
Connection engine	Generate feature-level candidate relationships	Rules first; ClaimGraph support; optional model rerank.
Review service	Accept/reject/supersede findings and control exports	Workflow tables, comments, assignments and signed decisions.
LLM gateway	Grounded summaries, explanations and report drafting	Approved local/private models; retrieval trace injected as citations.
Exchange gateway	Prepare, sanitize, approve, publish and revoke indicators	STIX/TAXII adapters, TLP policy and partner registry.

Reference implementation stack

A practical first implementation can use Next.js and TypeScript for the investigator application and API boundary; a Node service for `@knolo/core`, pack building and query orchestration; `@knolo/agents` for typed graph construction and client integration; the native Rust agent runtime for authoritative scheduling and policy enforcement; Python workers for parsing, evaluation and authorized model research; PostgreSQL for structured case state and event indexes; S3-compatible storage for evidence, packs and checkpoints; KMS or Vault for envelope encryption; and OpenTelemetry-compatible logs and traces.

This stack is not mandatory. The invariant is more important: original evidence and authorization live outside Knolo; knowledge and authority packs remain distinct; graph, policy and implementation versions are pinned; pack generation is reproducible; and every retrieval, tool call, model call or export passes through a policy-enforcing host boundary.

Deployment modes

Mode	Use case	Constraints
Managed private cloud	Nonprofit or enterprise teams needing the fastest operation	Dedicated tenant keys, private model endpoints and strict operational controls.
Sovereign/private VPC	Public-sector, regulated or regional data-residency deployments	Customer-managed keys, restricted egress, regional backups and audited support.
Offline workstation / enclave	Sensitive investigations, field work or disconnected analysis	Local evidence vault, Rust/Node runtime, signed pack import/export and manual synchronization.
Partner exchange node	Receive and publish sanitized indicators only	No raw case packs; STIX/TAXII, TLP and partner-specific policy.

7 Pack and data design

Pack boundaries should reflect authorization and lifecycle—not convenience. CCRNet must distinguish Knolo Core knowledge packs from Knolo Agents authority packs. A knowledge pack is a queryable derivative linked to governed records; an authority pack is a least-authority declaration for a graph and its effects.

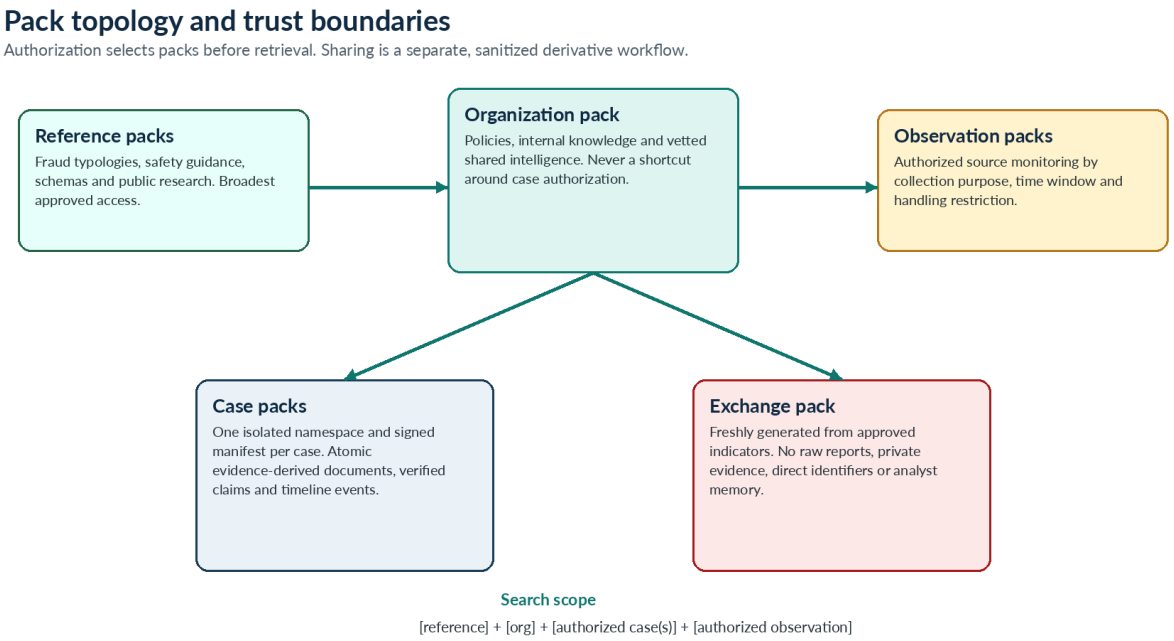

Figure 2. Proposed knowledge-pack topology. Case isolation remains the default; agent authority packs are bound separately to each workflow run; exchange is generated as a new derivative.

> Do not conflate the two pack types
> Knowledge packs answer “what content may be searched?” Authority packs answer “what may this graph do, in which namespace, with which tools and budget?” A run should record both identifiers and hashes.

Recommended pack classes

Pack class	Contents	Lifecycle
Reference pack	Fraud typologies, public safety guidance, schemas, known public infrastructure	Curated release; broad approved access.
Organization pack	Policies, internal procedures, shared vocabularies and verified organization intelligence	Versioned by governance team.
Case pack	Evidence-derived atomic documents, verified claims, events and case-local analyst conclusions	Immutable base + active LivePack; close with signed snapshot.
Observation pack	Authorized source captures and extracted signals for a defined collection purpose	Time-boxed; reviewed; separate from case until adopted.
Exchange pack	Approved, sanitized indicators and non-sensitive relationships	Generated per release; revocable; partner-scoped.

Pack class	Contents	Lifecycle
Evaluation pack	De-identified, purpose-approved benchmark examples and labels	Frozen dataset version with model card and access controls.

Atomic knowledge document model

Each Knolo document should represent one reviewable knowledge unit rather than an entire report. Atomic units improve retrieval precision, make provenance easier to display and let a reviewer correct one fact without rewriting unrelated content. Suggested document classes include source excerpts, timeline events, digital indicators, aliases, transactions, infrastructure observations, verified claims, analyst notes and typology references.

Illustrative BuildInputDoc

```
{
  id: "case:<case_uuid>:event:<event_uuid>",
  heading: "Payment request via messaging platform",
  namespace: "case.event.verified",
  text: [
    "Event type: payment_request.",
    "Occurred at: 2026-07-14T18:21:00Z.",
    "Channel: messaging_platform.",
    "Amount: 850 EUR.",
    "Indicator: wallet <tokenized_or_authorized_value>.",
    "Source artifact: artifact:<uuid>.",
    "Verification: analyst_confirmed."
  ].join(" ")
}
```

Stable identifiers and namespaces

Pattern	Example	Purpose
case:<case>:artifact:<id>	case:9f…:artifact:a1…	Evidence-derived excerpt or artifact description.
case:<case>:event:<id>	case:9f…:event:e4…	Atomic timeline event.
case:<case>:indicator:<type>:<id>	case:9f…:indicator:wallet:w7…	Case-scoped indicator record.
case:<case>:claim:<id>	case:9f…:claim:c3…	Human-verified claim with supporting evidence.
org:<org>:policy:<id>	org:hive:policy:p2…	Organization guidance or handling policy.
exchange:<release>:indicator:<id>	exchange:r18:indicator:d9…	Sanitized publishable derivative.

Metadata that stays outside the pack

- Original artifact location, content hash, MIME type, encryption key reference and malware status.
- Collection method, collector, authority or consent, jurisdiction, timestamp and source reliability.
- Data subject or victim handling restrictions, purpose, retention date and legal hold.
- Confidence, reviewer identity, verification history, correction history and dispute status.
- TLP label, export eligibility, partner scope, revocation status and disclosure log.
- Model versions, extraction prompts, transformation code version and evaluation status.

Pack manifest

Every pack should have a signed external manifest. The manifest should include pack ID and type, organization and case scope, Knolo/core version, schema version, source-record manifest hash, build timestamp, build pipeline version, document count, SHA-256 of the pack bytes, active key ID, signer identity, retention class and superseded pack ID. The Knolo patch base fingerprint should be preserved as a consistency field, while the CCRNet signature supplies trust and accountability.

> **Do not overload text metadata**
>
> Encoding every policy and provenance field into searchable text increases leakage and can distort ranking. Keep only the minimum retrieval-relevant labels in the document and resolve the stable ID to the protected metadata store after authorization.

8 Retrieval and connection model

Cross-case discovery should combine precise structured joins with Knolo's evidence retrieval—not force every problem through one ranking function.

Authorized multi-pack search

1. Authenticate the user and resolve organization, role, case membership, purpose and requested action.
2. Compile the allowed pack set: reference, organization, explicitly selected cases and authorized observation sources.
3. Run exact canonical indicator matches in the structured index.
4. Query each selected Knolo pack independently with the same normalized query and policy-approved options.
5. Capture pack ID, snapshot hash, options, latency, candidates and retrieval evidence in the query trace.
6. Merge candidates using a stable policy that preserves exact-match priority and never compares raw scores without calibration.
7. Resolve stable document IDs to authorized metadata and evidence cards.
8. Optionally apply bounded graph expansion or lexical-first semantic reranking and display those contributions separately.
9. Return suggestions—not conclusions—and require review for persistent case relationships or export.

Under Knolo Agents, retrieval should be exposed as an explicit, policy-gated host capability. The runtime request should carry the authorized knowledge snapshot set, query options and purpose. The result should persist ranked evidence, source identifier, locator and content hash—or an event reference to that immutable result—so replay can reuse recorded evidence without repeating external reads. [22]

Illustrative orchestration pseudocode

```
async function searchAuthorizedScope(request, principal) {
  const scope = await policy.authorizeSearch(principal, request);
  const exact = await indicatorIndex.match(scope, request.indicators);

  const traces = [];
  for (const packRef of stableSort(scope.packRefs)) {
    const pack = await packRegistry.mountVerified(packRef);
    const hits = query(pack, request.query, {
      topK: request.topKPerPack ?? 20,
      namespace: scope.namespaces,
      graph: { expand: request.graph === true, maxExtraTerms: 8 },
      semantic: approvedSemanticOptions(request, packRef)
    });
    traces.push({ packRef, hits });
  }

  return mergeWithEvidence({ exact, traces, policy: scope.mergePolicy });
}
```

Connection suggestion features

Feature	Interpretation	Default weight/handling
Exact canonical indicator	Same normalized wallet, hash, domain, account or address	Highest-confidence candidate; still verify source and context.
Alias or normalized token overlap	Same username, phone fragment, email local-part or phrase	Moderate; collision-aware and context-dependent.
Temporal proximity	Indicators or events occur within a bounded time window	Supporting feature only.
Co-occurrence pattern	Same group of indicators, channels or payment path	Stronger when repeated across independent artifacts.
ClaimGraph path	Bounded relationship or term expansion from pack graph	Lead feature; show path and evidence.
Lexical similarity	Narrative or tactic overlap with source citations	Useful for typologies; not identity resolution alone.
Semantic rerank	Reorders already grounded candidates	Optional; disclose model and score.
Analyst confirmation	Human accepts or rejects candidate with reason	Creates persistent reviewed relationship and evaluation label.

Suggestion lifecycle

State	Meaning	Allowed transition
Generated	System found a candidate relationship	To under_review, dismissed or expired.
Under review	Assigned analyst is examining evidence	To accepted, rejected, needs_more_evidence.
Accepted	Reviewer confirmed a defensible relationship	May be superseded or revoked with reason.
Rejected	Reviewer found collision, bad source or insufficient evidence	May be reopened only with new evidence.

State	Meaning	Allowed transition
Superseded	Newer reviewed relationship replaces it	Retained for audit.
Exchange-approved	Sanitized relationship approved for partner sharing	May be revoked; disclosure history preserved.

Scoring and calibration

Knolo scores should be interpreted inside the pack and query configuration that produced them. CCRNet should not simply sort raw scores from unrelated packs. A stable merge policy can create separate lanes—exact indicators, case-local lexical results, organization/reference results and expanded candidates—then rank within lanes and apply pack priority or calibrated percentile transforms. The UI should show the lane and evidence rather than a single opaque “risk score.”

9 Governed case workflow

The case lifecycle is the product. Knolo Agents can make the automated portions explicit and replayable, but preservation, qualified review and controlled transformation are what make the system trustworthy.

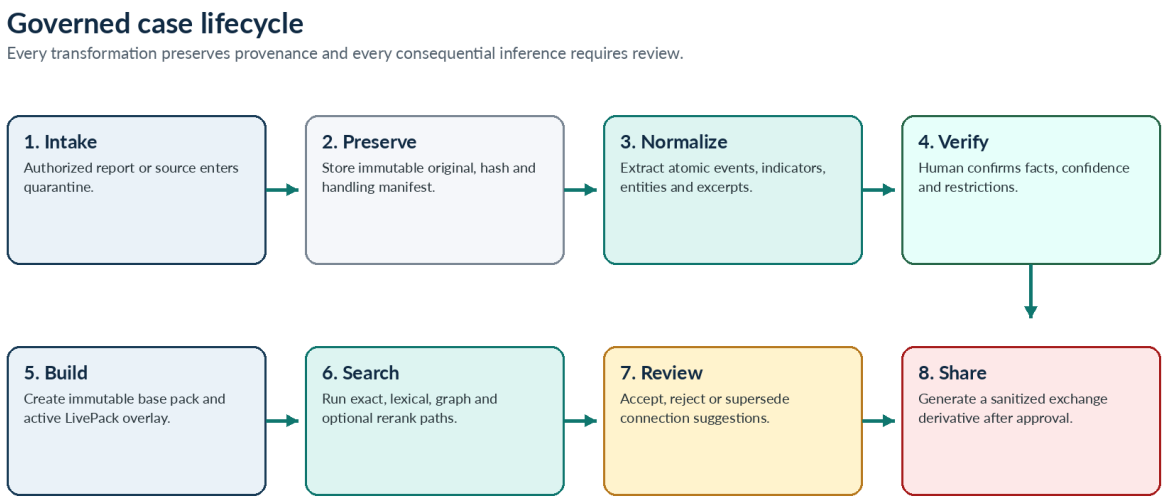

Figure 3. Governed lifecycle from intake to controlled sharing. Each automated stage may be a constrained graph; verification, persistent connections and release remain human or policy gates.

Agent-mediated detailed workflow

Stage	Agent or automated actions	Human or policy gate
Intake	Validate authorization, rate limits, file types and submission context	Accept source terms; select purpose and case ownership.

Stage	Agent or automated actions	Human or policy gate
Quarantine	Malware scan, content isolation, hash, timestamp and immutable write	Release artifact to processing or reject.
Normalize	Parse, OCR, canonicalize indicators, create excerpts and events	Confirm extraction quality and sensitive fields.
Verify	Propose claims, confidence and source relationships	Analyst confirms, edits or rejects each material fact.
Build	Compile verified atomic docs, generate pack and manifest, sign and register	Activate snapshot for case search.
Investigate	Policy-gated exact match and Knolo retrieval, bounded graph expansion, evidence cards and candidate suggestions	Reviewer accepts, rejects, supersedes or requests more evidence; agent cannot decide attribution.
Close/snapshot	Serialize overlay, freeze case version, retain audit and manifests	Case owner closes or reopens under policy.
Exchange	Constrained exchange-preparation graph sanitizes, maps schema, labels and checkpoints before suspension	Two-person approval or organization-defined equivalent resumes publication.

How LivePack should be used

At case creation, build a base pack from verified records. Mount it as read-only and create a LivePack for active edits. Use stable IDs for all documents. Batch approved changes into the overlay, serialize periodic snapshots, store the patch stream and signed manifest, and mark one snapshot as active. On case closure, serialize the final pack and freeze it; later corrections create a new signed version rather than rewriting history.

> **Memory is not evidence**
>
> Cortex is appropriate for analyst reminders, user preferences, session memory and provisional notes. A Cortex memory should not become a verified claim until it is linked to source evidence, reviewed and promoted through the structured case workflow.

Checkpoints, HITL and replay

Before a graph suspends for analyst review or an external dependency, CCRNet should persist an atomic checkpoint bound to the graph, authority pack, compiled policy, node implementation, state schema and contract hashes. Resume must fail closed when those artifacts change or the token expires. Ordered agent events support control-plane reconstruction, while the independent evidence audit log proves source handling and review history. The two records complement one another but are not interchangeable. [23–25]

Replay should default to verification or mocked effects. Repeating live effects—such as external model calls, partner publication or notification—must require fresh authorization and should never occur merely because a historical event log exists.

Observation workflow

Authorized Telegram, forum, marketplace or dark-web observations should enter purpose-specific observation packs—not flow directly into victim case packs. Collection records must state source, authorization, timestamp, acquisition method and handling restriction. Analysts may adopt an observation into a case only through an explicit evidence-linking action. This separation reduces contamination, preserves source context and simplifies deletion or policy changes.

10 Security, privacy and evidentiary integrity

The most important security decision is to keep sensitive originals and governance state outside portable search artifacts while ensuring the artifacts can be verified and revoked.

Security baseline

Control area	Minimum design
Encryption	Envelope encryption at rest; TLS in transit; separate keys by organization and optionally case; customer-managed keys for regulated deployments.
Identity	OIDC/SAML, MFA, short-lived service credentials, workload identity and no shared investigator accounts.
Authorization	Role + attribute + purpose checks; case membership; deny by default; policy before pack mount and export.
Network	Private subnets, restricted egress, model endpoint allowlist, administrative bastions and optional offline mode.
Application	Signed uploads, content-type verification, safe parsers, output encoding, secrets management and dependency scanning.
Audit	Append-only records for access, queries, pack builds, reviews, exports, policy changes, key use and deletion.
Monitoring	Anomaly alerts for bulk search, cross-case access, export spikes, failed policy checks and unusual model use.
Recovery	Encrypted backups, tested restore, key rotation, pack revocation and incident playbooks.

Evidence integrity

- Hash the original bytes at intake and every exported derivative.
- Record acquisition context, tool/version, collector, timestamp and authority or consent.
- Preserve the original artifact separately from extracted text and normalized records.
- Use deterministic transformation versions when possible and record nondeterministic model versions and prompts.
- Sign pack manifests and snapshots; store the signer and trusted timestamp.
- Never overwrite an accepted claim or review decision—supersede it with a reason.
- Keep the Knolo patch stream and the evidentiary audit log separate; both are useful, but they prove different things.

Privacy model

The product should follow purpose limitation, data minimization, accuracy, storage limitation, security and accountability principles. [17] A report may be lawful to retain for victim support but not lawful or appropriate to

publish, train on or expose to another tenant. CCRNet therefore needs separate permissions for collection, investigation, model evaluation and exchange rather than one broad consent flag.

Data class	Default handling	Exchange eligibility
Victim/contact data	Encrypted, case-restricted, minimized and retention-bound	No, except explicit authorized contact coordination.
Raw report narrative	Case-restricted with source reliability and dispute state	No.
Original evidence	Immutable vault, least privilege, malware-safe viewing	No.
Direct personal identifier	Tokenized or masked in most UI contexts	Normally no; exceptional legal workflow only.
Technical indicator	Case-restricted until verified and classified	Potentially, after sanitization and approval.
Typology/pattern	Aggregated and de-identified	Yes, when re-identification risk is controlled.
Analyst memory	Private or team-scoped; not treated as fact	No.

Agent authority and runtime security

- Agent definitions, pack bytes and retrieved artifacts should be treated as untrusted inputs until validated. The trusted boundary is the compiler/runtime plus host authorization and effect implementations.
- Credentials must never be serialized into authority packs, event logs or checkpoints. Tool implementations are host-owned and must validate arguments again at the effect boundary.
- Every effect should be denied by default unless the compiled policy grants the capability, namespace, tool and remaining budget. Retrieved text cannot grant authority or alter a graph.
- Checkpoint and replay integrity strengthens auditability but does not independently establish chain of custody, source authenticity or correctness of an analyst decision.
- Multi-agent handoffs must narrow or preserve authority; a child graph must never receive broader scope than the parent or requesting principal.

AI and prompt-injection controls

Cybercrime artifacts are hostile inputs. Messages, HTML, PDFs and source captures may contain instructions intended to manipulate an AI system. Extraction and summarization workers should treat artifact content as untrusted data, separate system instructions from evidence, disable tool execution by default, allowlist tools, cap context, sanitize active content and require citation-backed outputs. NIST's AI RMF and generative AI profile provide a useful governance structure for mapping, measuring, managing and governing these risks. [14,15]

- No model receives more case scope than the requesting principal is authorized to search.
- No retrieved text may change tool permissions, system policy or pack scope.
- Generated claims remain provisional until a reviewer links them to evidence.
- Embedding or rerank services must be approved for the data classification in use.
- Model logs must not retain raw evidence unless explicitly governed and encrypted.

Internet Computer guidance

Recommended boundary

The current Knolo ICP path exposes public search and controller-operated pack updates, with a documented pack-size cap. It is therefore unsuitable for raw CCRNet case evidence in its present form. Consider it only for

public reference packs, published safety material, sanitized indicators explicitly approved for public access, or non-sensitive integrity demonstrations.

11 Controlled exchange and interoperability

Exchange should be a policy-controlled publication process, not federated access to private case packs.

Exchange pipeline

1. Select an accepted indicator or relationship with supporting evidence and an authorized sharing purpose.
2. Generate a new exchange candidate that excludes raw narratives, victim data, analyst memory and private artifact locations.
3. Canonicalize values, apply minimization or keyed pseudonymization and assess re-identification risk.
4. Map eligible fields to an interoperability schema, attach confidence and handling labels, and specify validity or expiration.
5. Run automated policy checks and a human approval workflow; require two-person approval for high-impact releases.
6. Publish to a partner-scoped feed or package; record recipient, version and disclosure basis.
7. Support correction, expiration and revocation; notify recipients when policy requires.

STIX/TAXII role

STIX 2.1 provides a standard language for cyber threat and observable information, while TAXII 2.1 provides an application-layer protocol for exchanging that information. [10,11] CCRNet should use a constrained STIX profile for sanitized indicators, infrastructure, attack patterns, observed data and relationships. It should not force victim-support records, legal basis, consent, evidence chain or internal review workflow into STIX; those remain in the CCRNet data model.

CCRNet concept	Possible STIX object	Notes
Verified technical indicator	Indicator	Include pattern, validity, confidence and handling extension.
Observed occurrence	Observed Data	No direct victim identifiers; use minimal supporting observables.
Malicious infrastructure	Infrastructure	Publish only after verification and policy review.
Fraud or attack typology	Attack Pattern	Use CCRNet vocabulary mappings and references.
Relationship	Relationship	State confidence and provenance summary; avoid implying attribution beyond evidence.
Case or victim report	Not directly exported	Keep private; exchange only derived objects.

Handling labels

Use FIRST Traffic Light Protocol 2.0 labels—TLP:RED, TLP:AMBER, TLP:GREEN and TLP:CLEAR—where appropriate, combined with contractual partner restrictions and jurisdiction-specific policy. [16] TLP is a sharing signal, not an authorization engine; the exchange gateway must still enforce recipient, purpose, expiration and revocation.

Pseudonymization

For private partner correlation, CCRNet can generate keyed HMAC tokens for selected normalized identifiers. A keyed construction avoids the enumeration weaknesses of plain hashes for predictable values such as phone numbers or email addresses. Keys should be partner- or consortium-specific, rotated, access-controlled and never used as a substitute for legal or ethical review. Public releases should generally avoid stable person-linked tokens altogether.

12 API and service boundary

The public developer preview can evolve into a resource-oriented API that separates evidence, packs, search, review and exchange.

Proposed API surface

Endpoint	Purpose	Key controls
POST /v1/cases	Create a case and assign scope, purpose and retention	Organization role, purpose and policy validation.
POST /v1/cases/{id}/artifacts	Create signed upload and register artifact	Case write access, file policy and consent/legal basis.
POST /v1/cases/{id}/normalize	Queue extraction and normalization	Artifact release from quarantine; approved workers.
POST /v1/cases/{id}/snapshots	Build, sign and activate a Knolo snapshot	Verified records only; manifest and audit.
POST /v1/search	Search authorized packs and exact indicator index	Explicit case scope, purpose, trace ID and result limits.
GET /v1/search/{traceId}	Retrieve reproducible query trace	Same or stronger authorization than original request.
GET /v1/connections/{id}	View suggestion and supporting evidence	Authorized case access.
POST /v1/connections/{id}/reviews	Accept, reject or request evidence	Reviewer role, reason code and immutable decision record.
POST /v1/exchange/candidates	Prepare sanitized derivative	Accepted source relationship and export permission.
POST /v1/exchange/{id}/approve	Approve and publish partner object	Two-person or configured approval; TLP and recipient scope.
POST /v1/exchange/{id}/revoke	Revoke or correct published object	Authorized governance role; recipient notification log.

Search request example

Illustrative request

```
POST /v1/search
{
  "query": "same wallet and delivery-deposit phrasing",
  "scope": {
    "organizationId": "org_hive",
    "caseIds": ["case_01", "case_19"],
    "includeReference": true,
    "includeObservations": false
  },
  "retrieval": {
    "exactIndicators": true,
    "lexical": true,
    "graphExpansion": true,
    "semanticRerank": false,
    "topKPerPack": 20
  },
  "purpose": "authorized_case_investigation"
}
```

Result contract

Illustrative response

```
{
  "traceId": "trace_7f...",
  "snapshotSetHash": "sha256:...",
  "results": [{
    "resultId": "result_a1...",
    "lane": "exact_indicator",
    "packId": "case_01_snapshot_12",
    "documentId": "case:01:indicator:wallet:w7",
    "namespace": "case.indicator.verified",
    "scores": {
      "lexical": 4.81,
      "semantic": null,
      "mergeRank": 1
    },
    "evidenceRefs": ["artifact:93...#excerpt:4"],
    "reviewState": "unreviewed"
  }]
}
```

Agent run endpoints

Endpoint	Purpose
POST /v1/agent-runs	Start a pinned graph with organization, purpose, authority-pack hash, knowledge-snapshot set and initial state.
GET /v1/agent-runs/{id}/events	Read ordered, redacted control-plane events and evidence-result references.
GET /v1/agent-runs/{id}/checkpoint	Read checkpoint metadata and compatibility hashes, not host credentials.
POST /v1/agent-runs/{id}/resume	Resume a valid HITL suspension with typed reviewer input and fresh authorization.
POST /v1/agent-runs/{id}/cancel	Terminate a run, revoke pending tokens and record the reason.

Service contracts

- Every request carries principal, organization, purpose and trace context.
- Every pack mount verifies registry status, manifest signature, checksum, scope and revocation.
- Every result resolves through a stable document ID; raw pack text is not the sole source of metadata.
- Every persistent connection records the exact snapshot set and query trace that generated it.
- Every export records transformation, approvers, recipient, handling label and revocation path.

13 Product surfaces and deployment modes

The user experience should make trust boundaries visible rather than hide them behind a single chat interface.

Investigator workspace

Surface	Primary functions
Case overview	Purpose, status, ownership, retention, risk flags and recent activity.
Evidence	Original artifacts, safe preview, hashes, provenance, extraction status and access history.
Timeline	Verified events with source links, confidence and correction history.
Indicators	Canonical values, occurrences, validation state and exact cross-case matches.
Search	Explicit scope selector, lexical/graph settings, query trace and evidence cards.
Connections	Generated suggestions, feature explanation, reviewer queue and decisions.
Reports	Grounded summaries and export packages with citations and approval.
Audit	Who accessed, searched, changed, reviewed or exported what and why.

Avoid a chat-only design

A conversational assistant can help formulate queries and summarize retrieved evidence, but it should sit beside structured evidence, timeline, indicator and connection views. Investigators need to compare artifacts, inspect provenance, filter by case scope and record decisions. The interface should show exactly which packs are active and make it difficult to accidentally broaden the search boundary.

Access profiles

Profile	Access	Primary use
Community support	Own report or consented support case; educational reference packs	Victim guidance, preservation checklist and safe referral.
Researcher	De-identified evaluation packs and approved public/partner data	Method evaluation and typology research.
Investigator	Assigned cases, organization pack and approved observations	Evidence-led case work and connection review.
Governance reviewer	Policy, audit, retention, export and correction workflows	Oversight and accountability.

Profile	Access	Primary use
Exchange partner	Partner-scoped sanitized indicators only	Defensive correlation and feedback.
Platform administrator	Infrastructure and tenant configuration, not automatic case content	Operations with separation of duties.

14 Phased delivery roadmap

Delivery should advance only when governance and quality gates are met. The roadmap below is an illustrative sequence, not a launch commitment.

Phase	Scope	Exit gate
0 — Foundations	Threat model, lawful-use policy, data classification, source rules, retention, schemas, agent graph conventions, authority-pack model, tool contracts, test corpus and architecture decisions.	Approved governance baseline; synthetic/de-identified test data; security review.
1 — Isolated case alpha	Identity, case intake, evidence vault, normalization, case packs, scoped search, evidence cards, constrained intake/search graphs, checkpoints and audit.	No unauthorized cross-case access; reproducible builds; acceptable retrieval recall on benchmark.
2 — Cross-case beta	Exact indicator index, multi-pack orchestrator, connection suggestions, review queue, HITL resume, LivePack snapshots and deterministic replay.	False-connection targets met; reviewer workflow and trace reconstruction pass.
3 — Partner exchange pilot	Sanitization, STIX/TAXII mapping, TLP, approval, partner feeds, correction and revocation.	Privacy review; partner agreements; disclosure simulation and revocation drill.
4 — Model lab	De-identified evaluation packs, retrieval experiments, entity-resolution research and typology models.	Model cards, drift tests, safety thresholds and no consequential automation.
5 — Institutional scale	Multi-region/private deployments, offline nodes, advanced connectors and operational hardening.	External security assessment, restore tests, observability and support readiness.

Build versus extend

Use as-is	Extend Knolo where generally reusable	Build in CCRNet
Pack build/mount/query	Structured optional metadata section	Identity, organizations and case authorization
Namespaces and source IDs	Pack signature/manifest helpers	Evidence vault and chain of custody
Lexical and optional rerank	Multi-pack query helper with trace output	Normalization and indicator canonicalization
ClaimGraph expansion	Hierarchical namespace conventions	Exact indicator index
LivePack and patch packs	Batch LivePack mutation API/performance work	Review, workflow and audit

Use as-is	Extend Knolo where generally reusable	Build in CCRNet
Cortex memory	Provenance adapters or typed document conventions	Sanitization and exchange
Node/Python/Rust runtimes	Cross-runtime conformance suite	LLM gateway and product UI

Engineering priorities for Knolo integration

1. Publish a CCRNet document convention and validation package on top of `BuildInputDoc`.
2. Create a signed pack-manifest utility that records source manifest hash and builder version.
3. Implement an authorization-aware pack registry and local cache with revocation.
4. Build deterministic multi-pack fanout and merge with complete query traces.
5. Add domain-specific retrieval benchmarks for wallets, domains, aliases, scam phrasing and typologies.
6. Batch or queue LivePack changes and measure rebuild behavior on realistic case sizes.
7. Create cross-runtime golden fixtures for Node, Python and Rust query parity.
8. Publish a CCRNet agent-graph package with pinned state schemas, tool contracts and least-authority pack templates.
9. Implement an atomic checkpoint and ordered-event store with redaction, retention and compatibility-hash validation.
10. Build HITL review/resume flows and replay tests that distinguish verify-only, mocked-effect and live-effect authorization.

15 Evaluation framework

A cybercrime intelligence product should be evaluated on retrieval, connection quality, human usefulness, privacy and operational trust—not chatbot fluency.

Retrieval evaluation

Metric	Question	Required slices
Recall@K	Did the authorized search retrieve all known relevant evidence?	Indicator type, typology, language, source and case size.
Precision@K	How much reviewer time is spent on irrelevant results?	Exact vs lexical vs graph vs semantic lane.
MRR/nDCG	How early do the strongest evidence-backed results appear?	Case-local, cross-case and reference-pack queries.
Trace reproducibility	Can the same snapshot set reconstruct the same result order?	Runtime, platform and version.
Citation completeness	Does every claim resolve to authorized evidence?	Summary, connection and export surfaces.
Latency and memory	Can the selected deployment meet interactive use?	Pack count, pack size, cold/warm cache and offline mode.

Connection evaluation

- False connection rate by feature type and indicator collision class.
- Reviewer acceptance, rejection and “needs more evidence” rates.
- Time from report intake to first defensible cross-case link.
- Percentage of accepted links supported by two or more independent artifacts.
- Reversal and supersession rate after later evidence.
- Disparity in false positives across languages, regions, source types and report completeness.

Governance and safety metrics

Metric	Target direction
Unauthorized pack mounts or cross-case exposures	Zero; fail closed.
Queries without purpose and trace context	Zero.
Exports without approval and recipient record	Zero.
Evidence cards missing provenance	Zero for persistent findings.
Retention or deletion actions not propagated to derived packs	Zero unresolved exceptions.
Public or partner corrections completed within policy window	Increasing compliance; measure median and tail.
Model outputs adopted without human evidence review	Zero for consequential findings.

Benchmark strategy

The Knolo repository reports strong recall and ranking metrics on its March 2026 benchmark. [5] CCRNet should treat that as an engine baseline, not product validation. Build a domain benchmark from synthetic and governed de-identified examples with explicit relevance judgments, exact-indicator cases, hard alias collisions, multilingual phrasing, incomplete reports and adversarial noise. Freeze pack versions and publish test methodology internally so changes to tokenization, graph extraction, query expansion or reranking can be compared.

16 Risks and decisions

The largest risks are not whether search returns text. They are unauthorized scope, false identity linkage, provenance loss, unsafe collection and inappropriate sharing.

Principal risks

Risk	Failure mode	Mitigation
False entity resolution	Two people or operations share an alias, phrase or infrastructure	Exact/canonical lanes, collision estimates, multiple evidence features and mandatory review.
Sensitive-data leakage	Raw report content enters logs, models or exchange	Data classification, redaction, private endpoints, logging controls and derivative exchange.
Scope bypass	Search or model call reaches unauthorized packs	Policy before mount, deny-by-default registry and audited trace.
Source contamination	Unverified observation becomes case fact	Separate observation packs and explicit adoption workflow.
Prompt injection	Hostile evidence changes model behavior or tool calls	Treat content as data, tool allowlists, instruction separation and citation validation.

Risk	Failure mode	Mitigation
Pack drift	Active pack no longer matches source records or policy	Signed manifest, source hash, version registry, revocation and rebuild checks.
Score misuse	Opaque ranking becomes a guilt or risk score	Feature explanation, lane labels, no person score and human review.
Exchange re-identification	Sanitized indicators combine to reveal a victim or subject	Risk review, minimization, partner-specific tokens, expiration and revocation.
Unsafe collection	Source monitoring crosses legal or ethical boundary	Approved source registry, purpose, jurisdiction review and no criminal participation.
Operational overreach	Research prototype is presented as proven enforcement intelligence	Clear status, controlled pilots, evaluation and external review.

Decisions required before implementation

- Who is the first authorized user: internal researchers, nonprofit investigators, private fraud teams or institutional partners?
- Which jurisdictions and legal bases govern intake, observation, retention and exchange?
- Will deployment begin as a managed private tenant, customer VPC or offline application?
- Which source types are allowed in phase one, and which require separate approval?
- What is the minimum evidence threshold for an accepted connection?
- Who may approve partner exchange and how are corrections or revocations delivered?
- Which models, if any, may process each data classification?
- Will richer pack metadata be added to Knolo core or maintained only in CCRNet manifests?

High-value technical experiments

Experiment	Decision it informs
Case pack size and mutation benchmark	Whether LivePack is sufficient for active case write patterns or needs batching/index deltas.
Multi-pack score calibration study	How to merge exact, lexical, graph and semantic lanes without misleading ranking.
Indicator collision corpus	Which normalized identifiers are safe to treat as high-confidence links.
Prompt-injection red-team corpus	Which parsing and model boundaries are required for hostile evidence.
Cross-runtime golden query suite	Whether Node, Python and Rust can reconstruct equivalent results.
Sanitization re-identification test	Which fields and combinations may enter partner exchange.

17 Recommended MVP

The first credible product is not a global cybercrime graph. It is a secure, evidence-led case workspace that proves isolation, reproducible search and reviewable connections.

MVP scope

Capability	Included	Deferred
Tenancy	One organization with admin, investigator and reviewer roles	Cross-organization federation.
Intake	Manual form, JSON/CSV and controlled document upload	Open public submissions and automated source crawling.
Evidence	Encrypted object vault, hashing, quarantine, safe preview	Advanced forensic acquisition tooling.
Knowledge	Case and reference packs; immutable snapshots; LivePack working overlay	Large-scale streaming packs.
Search	Exact indicators, lexical query, namespace filters and optional bounded graph expansion	Default semantic rerank and opaque global ranking.
Agents	Pinned intake, search, review-context and exchange-preparation graphs with least authority, checkpoints, HITL and replay	Unsupervised autonomous investigation, self-approval or unbounded external action.
Connections	Evidence-backed suggestions and human accept/reject workflow	Automated identity attribution.
AI	Grounded summary/report drafting with exact citations in private boundary	Person-level prediction or autonomous action.
Audit	Access, query, build, review and export logs	Full external transparency portal.
Exchange	Export a reviewer-approved internal package	Partner TAXII federation and public feeds.

MVP acceptance criteria

- An investigator cannot query or resolve a document from a case they are not assigned to.
- A reviewer can reconstruct a result from the stored snapshot set, query and options.
- Every persistent connection displays supporting evidence and review history.
- Deleting or restricting a source record revokes or rebuilds affected derived packs under policy.
- No raw evidence reaches an unapproved external model endpoint.
- Pack corruption, signature failure or revocation causes mount to fail closed.
- The domain benchmark meets agreed recall and false-connection thresholds before pilot use.

First implementation backlog

1. Define the case, artifact, event, indicator, claim, review and pack-manifest schemas.
2. Build organization/case authorization and an append-only audit outbox.
3. Create the encrypted evidence vault and quarantine pipeline.
4. Implement deterministic normalization for initial indicator types: wallet, domain, URL, email, phone and account handle.
5. **Create the CCRNet document compiler and signed knowledge-pack registry around `@knolo/core`.**
6. **Define CCRNet agent graphs, tool contracts and least-authority packs using `@knolo/agents`.**
7. **Build the authorized search host with exact and Knolo lanes, then expose it as a policy-gated retrieval effect.**

8. **Implement evidence cards, checkpointed HITL review and the connection decision queue.**
9. **Add a durable ordered-event/checkpoint store and verify-only/mocked-effect replay tests.**
10. **Add private grounded report drafting only after citation completeness tests pass.**
11. **Run security, privacy, retrieval, agent-policy and false-link evaluations on synthetic/de-identified cases.**

> **Launch gate**
>
> Do not open public report intake or partner exchange until the isolated case product has passed authorization testing, evidence-trace reconstruction, retention/deletion tests, prompt-injection testing and a privacy review.

18 Conclusion

The best use of Knolo in CCRNet is precise: use Knolo Core to make governed knowledge portable and deterministic, and use Knolo Agents to make workflow execution explicit, least-authority and replayable—without making either layer responsible for evidence, identity or final judgment.

CCRNet has a credible product thesis: private case intelligence that helps authorized people find evidence-backed recurring patterns while preserving human review and controlled sharing. Knolo Core provides the retrieval primitives—isolated artifacts, lexical-first search, filtering, graph expansion, active overlays and portable runtimes. Knolo Agents provides typed graphs, least-authority policy, host-effect boundaries, checkpoints, HITL and replay. [1–8,20–26]

The product succeeds only if the surrounding architecture is equally deliberate. Original evidence must remain in an encrypted system of record; authorization must select knowledge scope and compile agent authority before execution; rich provenance and review state must remain outside the current four-field document model; host effects and credentials must stay outside packs; exact indicator correlation must complement textual search; and every persistent connection or exchange action must remain explainable, reviewable and reversible.

The recommended next step is a single-organization investigator workspace and reference implementation. Prove case isolation, reproducible search, constrained agent authority, checkpointed review, evidence traceability and safe replay first. Then add controlled observation, partner exchange and model research as separately governed modules. That sequence turns the CCRNet vision into a defensible public-interest platform rather than an over-broad cybercrime database or autonomous accusation system.

> **Final positioning**
>
> CCRNet is the governed cybercrime intelligence product. Knolo is the deterministic knowledge substrate inside it. The evidence vault, control plane, review workflow and exchange policy are what make the combined system safe and operationally useful.

Appendix A — Detailed Knolo Core / Knolo Agents to CCRNet mapping

Repository behavior	CCRNet mapping	Implementation note
`buildPack()` stores normalized blocks, doc IDs, namespaces, headings and optional graph/semantic sections.	Compile verified atomic case records into immutable query artifacts.	Keep rich metadata in signed external manifest and database.
Lexical query supports phrases, namespace/source filters, query expansion and graph expansion.	Case search, typology search and evidence retrieval.	Policy selects packs before query; save options in trace.
Semantic rerank operates on lexical candidates and can expose evidence values.	Optional ordering improvement for hard language matches.	Approved data boundary and model required; never sole identity signal.
LivePack uses stable IDs, overlay upserts and tombstones, then materializes a merged pack.	Active case working set.	Batch writes; snapshot reviewed states; benchmark rebuild cost.
Patch packs include actor, timestamp, upsert/remove operations and base fingerprint.	Incremental synchronization or update package.	Add signer identity, trusted time and independent audit.
ClaimGraph uses deterministic node/edge identifiers and bounded expansion.	Candidate relationship and query expansion.	Preserve evidence path; do not equate graph edge with verified relationship.
Cortex provides append-only memory and consolidation back to docs.	Analyst memory, session context and provisional notes.	Promotion to case fact requires evidence and review.
Node/browser, Python and Rust runtimes.	Web service, research jobs, offline/edge nodes.	Maintain golden conformance fixtures.
ICP canister offers public lexical search and controller pack updates.	Public reference or approved sanitized content only.	Do not use for private case data in current design.

Appendix B — Proposed record schemas

Pack manifest

Illustrative TypeScript

```
type CCRNetPackManifest = {
  manifestVersion: 1;
  packId: string;
  packType: "reference" | "organization" | "case" | "observation" | "exchange" | "evaluation";
  organizationId: string;
  caseId?: string;
  knoloCoreVersion: string;
  documentSchemaVersion: string;
  sourceManifestHash: string;
  packSha256: string;
  baseFingerprint?: string;
  documentCount: number;
  builtAt: string;
  builderVersion: string;
  signerKeyId: string;
  signature: string;
  retentionClass: string;
  supersedes?: string;
  status: "active" | "superseded" | "revoked";
};
```

Provenance record

Illustrative TypeScript

```
type ProvenanceRecord = {
  recordId: string;
  artifactId: string;
  collection: {
    method: string;
    collectedAt: string;
    collector: string;
    authorityOrConsentRef: string;
    sourceUri?: string;
  };
  integrity: {
    originalSha256: string;
    mimeType: string;
    sizeBytes: number;
    storageObjectId: string;
  };
  handling: {
    classification: string;
    purpose: string[];
    retentionUntil?: string;
    legalHold?: boolean;
    tlp?: "RED" | "AMBER" | "GREEN" | "CLEAR";
  };
  transformations: Array<{
    operation: string;
    toolVersion: string;
    performedAt: string;
    outputHash: string;
  }>;
};
```

Connection suggestion

Illustrative TypeScript

```
type ConnectionSuggestion = {
  suggestionId: string;
  caseIds: string[];
  snapshotSetHash: string;
  queryTraceId: string;
  features: Array<{
    type: "exact" | "canonical" | "lexical" | "graph" | "semantic" | "temporal";
    value: number;
    explanation: string;
    evidenceRefs: string[];
  }>;
  state: "generated" | "under_review" | "accepted" | "rejected" | "superseded";
  reviews: Array<{
    reviewerId: string;
    decision: string;
    reasonCode: string;
    decidedAt: string;
  }>;
};
```

Appendix C — Governance checklist

Before a new source or feature is enabled	Evidence required
Lawful and ethical authority	Documented purpose, jurisdiction review, source terms and collection limits.
Data classification	Fields, sensitivity, victim impact, direct identifiers and model eligibility.

Before a new source or feature is enabled	Evidence required
Authorization design	Roles, attributes, case scope, administrator separation and break-glass procedure.
Retention and correction	Retention schedule, legal holds, deletion propagation and dispute workflow.
Retrieval evaluation	Domain test set, relevance labels, false-link analysis and regression thresholds.
AI safety	Approved endpoint, data policy, prompt-injection tests, citation requirement and fallback.
Exchange review	Sanitization test, TLP/recipient policy, approval, expiration and revocation drill.
Operational readiness	Backups, restore, key rotation, monitoring, incident response and support ownership.

References

1. CCRNet — Home and program overview
https://www.ccrnet.org/

2. CCRNet — Research
https://www.ccrnet.org/research/

3. CCRNet — Platform preview
https://www.ccrnet.org/platform/

4. CCRNet — Developers and proposed architecture
https://www.ccrnet.org/developers/

5. Knolo — Public product and documentation site
https://www.knolo.dev/

6. HiveForensics-AI/knolo-core — @knolo/core README and builder implementation
https://github.com/HiveForensics-AI/knolo-core/tree/main/packages/core

7. knolo-core — Query implementation
https://github.com/HiveForensics-AI/knolo-core/blob/main/packages/core/src/query.ts

8. knolo-core — LivePack and patch-pack implementations
https://github.com/HiveForensics-AI/knolo-core/blob/main/packages/core/src/live.ts

9. FBI Internet Crime Complaint Center — 2025 Internet Crime Report
https://www.ic3.gov/AnnualReport/Reports/2025_IC3Report.pdf

10. OASIS — STIX Version 2.1
https://docs.oasis-open.org/cti/stix/v2.1/stix-v2.1.html

11. OASIS — TAXII Version 2.1
https://docs.oasis-open.org/cti/taxii/v2.1/taxii-v2.1.html

12. OASIS Cyber Threat Intelligence TC — current STIX/TAXII work and errata
https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cti

13. Europol — Internet Organised Crime Threat Assessment 2026
https://www.europol.europa.eu/publication-events/main-reports/iocta-2026-evolving-threat-landscape

14. NIST — Artificial Intelligence Risk Management Framework (AI RMF 1.0)
https://www.nist.gov/itl/ai-risk-management-framework

15. NIST — Artificial Intelligence Risk Management Framework: Generative AI Profile (NIST AI 600-1)
https://www.nist.gov/publications/artificial-intelligence-risk-management-framework-generative-artificial-intelligence

16. FIRST — Traffic Light Protocol 2.0
https://www.first.org/tlp/

17. European Commission — Principles of the GDPR
https://commission.europa.eu/law/law-topic/data-protection/rules-business-and-organisations/principles-gdpr_en

18. NIST — Cybersecurity Framework 2.0
https://www.nist.gov/cyberframework

19. NIST — Privacy Framework
https://www.nist.gov/privacy-framework

20. HiveForensics-AI/Knolo-Agents — Project guide and architecture
https://github.com/HiveForensics-AI/Knolo-Agents

21. Knolo Agents — @knolo/core boundary
https://github.com/HiveForensics-AI/Knolo-Agents/blob/main/docs/core-boundary.md

22. Knolo Agents — Retrieval contract and provenance
https://github.com/HiveForensics-AI/Knolo-Agents/blob/main/docs/retrieval.md

23. Knolo Agents — Policy enforcement and host-owned effects
https://github.com/HiveForensics-AI/Knolo-Agents/blob/main/docs/policy-enforcement.md

24. Knolo Agents — Checkpoints and human-in-the-loop resume
https://github.com/HiveForensics-AI/Knolo-Agents/blob/main/docs/checkpoints.md

25. Knolo Agents — Deterministic replay modes
https://github.com/HiveForensics-AI/Knolo-Agents/blob/main/docs/replay.md

26. @knolo/agents package metadata — version 0.1.3 and @knolo/core peer boundary
https://github.com/HiveForensics-AI/Knolo-Agents/blob/main/packages/agents/package.json

Public sources and repository main branches were reviewed for this publication on 4 August 2026. Implementation must pin and revalidate exact package, commit and contract versions.