ODA3 INSTITUTE

Practitioner Cheat Sheet

Vector Database Security Checklist

DocID: ODA3-2026-07-CHT-SEC-012
Classification: Public — Practitioner Reference

Version 1.3 | Public | 17 July 2026

Publisher: ODA3 Institute

Publication type: Practitioner Cheat Sheet

Review status: Publication edition

How to use this checklist: Define the approved vector-data boundary; secure ingestion, embedding, storage, retrieval and deletion as one lifecycle; test authorization at query time; preserve provenance and version evidence; monitor for poisoning, leakage and abnormal access; and route material events through the organization's AI incident process.

> **Key definition — secure vector retrieval:** A retrieval operation is secure only when the requesting identity is authenticated, authorization is enforced against authoritative policy at query time, the returned objects remain within the approved tenant and data boundary, provenance is reconstructable, and the result is suitable for the downstream AI use case.

1. Executive Overview

Vector databases are frequently treated as interchangeable search infrastructure. In an AI system they can become part of the decision boundary: retrieved content changes what a model sees, what it says and—when connected to agents—what it does. A conventional database control baseline remains necessary, but it does not by itself address embedding-model changes, similarity-based retrieval, poisoned knowledge, stale authorization metadata, cross-tenant nearest-neighbour results or indirect instructions embedded in retrieved content.

The material risks are not limited to direct database compromise. An authorized contributor may ingest malicious or misleading documents; an application may omit a tenant filter; an embedding migration may silently change retrieval behaviour; deleted source data may remain in an index or backup; and an attacker may infer sensitive membership or content through repeated similarity queries. Because embeddings can encode information derived from protected source material, classifying vectors as harmless numerical data is unsafe.

This checklist uses the ODA3 operating loop. **GAISSF™** supplies governance, ownership, control and evidence expectations for the vector-data lifecycle. **UAIF™** structures the identity, classification, impact, causality and evidence of a vector-related incident. **AI-IRF™** routes containment, eradication, recovery and lessons learned. Verified findings return to GAISSF™ so ingestion, retrieval and assurance controls change across the affected portfolio.

NIST AI RMF and its Generative AI Profile address governance, measurement, data provenance, value-chain and retrieval-augmented-generation risks. NIST's adversarial machine-learning taxonomy covers poisoning, privacy and evasion concepts. OWASP's 2025 LLM risks explicitly identify data/model poisoning and vector/embedding weaknesses. These sources provide external alignment; they do not establish equivalence, conformity or legal compliance.

Primary actors are the system owner, data owner, AI security architect, application engineer, platform/database operator, MLOps team, identity team, privacy/legal function, SOC/incident response and independent assurance. Each needs a defined decision right: who approves a collection, changes an embedding model, grants ingestion rights, authorizes retrieval, suspends refresh, restores an index and confirms deletion.

> **Notably Absent / Evidence Boundaries:** No control in this checklist proves that embeddings cannot reveal source information, that a similarity threshold is universally safe, or that a retrieved document is truthful because it is authorized. Vector products differ materially in isolation, filtering, encryption, consistency, backup and audit capabilities. This document does not provide product certification, a legal retention determination or a verified ODA3 control identifier. Exact GAISSF™, UAIF™ and AI-IRF™ internal identifiers were not available in the authoritative source set and are not inferred. Implementation reduces risk but does

not eliminate poisoning, inference, authorization or residual model-behaviour risk.

2. Key Concepts & Definitions

Term	Operational definition
Vector database	Data system optimized to store embeddings and retrieve objects using similarity or hybrid search, normally with metadata and source references.
Embedding	Numerical representation produced by an embedding model from text, image, audio or another object; it may retain information about its source.
Embedding model	Versioned model that maps source objects into a vector space. Changing it can change dimensions, distances, ranking and security behaviour.
Collection / index	Logical and physical grouping used to store and search vectors; names differ by product. It is not automatically a security boundary.
Namespace / partition	Product-specific subdivision used for organization or isolation; its enforcement properties must be tested rather than assumed.
Approximate nearest neighbour (ANN)	Search method that trades exactness for performance when locating similar vectors; index parameters can affect recall and exposure.
Distance metric	Similarity function such as cosine, dot product or Euclidean distance. Scores are model- and distribution-specific, not universal confidence values.
Metadata filter	Attribute predicate applied to narrow retrieval, commonly by tenant, owner, classification or document state. It is security-relevant only when enforced server-side and fail-closed.
Pre-filtering	Applying mandatory authorization predicates before or as part of candidate selection so unauthorized objects are outside the searchable result set for that request.
Post-filtering	Removing unauthorized objects only after search candidates have been returned to an application component; it can expose protected objects to that component and should not be treated as the primary authorization boundary.
Vector retrieval gateway	Trusted enforcement service that authenticates the caller, resolves current entitlements, constructs immutable mandatory filters, bounds disclosure and records the retrieval decision.
Hybrid search	Retrieval combining vector similarity with keyword, sparse or structured search; every branch must apply the same authorization policy.
Retrieval-augmented generation (RAG)	Pattern that retrieves external content and supplies it to a generative model as context. Retrieval improves grounding only within source and control limits.
Ingestion pipeline	Workflow that acquires, parses, classifies, chunks, embeds, approves and writes source material and metadata.
Chunk	Segment of a source object stored or retrieved independently. Chunking may break access, context, deletion and provenance assumptions.
Provenance	Evidence connecting a vector and chunk to its source, owner, ingestion event, transformations, approvals and versions.
Tenant isolation	Enforced prevention of one tenant’s identities, data and queries from accessing another tenant’s objects.
Query-time authorization	Evaluation of the requesting identity and current policy during each retrieval, not reliance on filters generated by the model or client alone.
Retrieval poisoning	Insertion or manipulation of content or metadata to influence what is retrieved and thereby affect downstream output or action.
Indirect prompt injection	Instructions embedded in retrieved content that attempt to redirect the model or agent beyond the intended task.

Embedding inversion	Attempt to reconstruct or infer information about source content from an embedding or related access.
Membership inference	Attempt to determine whether a particular record, concept or source is represented in a dataset or index.
Tombstone	Marker indicating logical deletion pending physical removal, compaction, backup expiry or downstream propagation.
Version-pinned retrieval bundle	Approved combination of embedding model, chunker, preprocessing, index, distance metric, filters, reranker and retrieval policy.
Retrieval trace	Record joining requester, query reference, policy decision, index/version, candidates, filters, scores, returned chunks and downstream use.
Ground-truth evaluation set	Controlled set of queries, authorized expected sources and negative cases used to measure retrieval and leakage behaviour.
Data residency	Requirement that data, derived representations, logs or backups remain in defined locations.
Evidence tier	T1 primary verified; T2 corroborated authoritative; T3 limited/single-source; T4 unverified or hypothesis.

3. Threat and Risk Landscape

3a. Actor taxonomy

Actor	Motivation / failure pattern	Capability	Likely vectors / exposure
External attacker	Exfiltration, influence, disruption or lateral movement	Medium–high	Stolen API key, exposed endpoint, query probing, injection, denial of wallet
Malicious contributor	Poison knowledge or establish persistence	Medium	Authorized upload, metadata manipulation, adversarial chunks, duplicate flooding
Privileged insider	Extract, alter or conceal data	High	Administrative export, filter bypass, deletion suppression, audit tampering
Application developer	Ships client-side or incomplete authorization	Medium	Missing tenant predicate, unsafe fallback, raw vector exposure
Data steward	Approves stale, misclassified or unlicensed material	Low–medium	Weak provenance, excessive retention, incorrect classification
Supplier / managed service	Configuration drift, platform compromise or insufficient evidence	High	Control-plane change, backup exposure, mutable service, log limitations
Automated ingestion source	Propagates malicious or erroneous content	Variable	Compromised crawler, poisoned feed, parser exploit, uncontrolled refresh
Model or agent	Generates unsafe filters or over-broad queries	Variable	Model-authored predicate, tool abuse, recursive retrieval, cross-domain query

3b. Vector and control-gap analysis

Vector / gap	Description	Prerequisite	Typical impact	Reference relationship
Missing tenant filter	Query searches a shared collection without authoritative tenant restriction	Shared index; client-generated filter	Cross-tenant disclosure	OWASP LLM08; conventional access control
Filter injection /	Attacker alters structured filter or exploits permissive	Unvalidated query DSL; dynamic filter	Unauthorized objects returned	OWASP LLM08

bypass	fallback			
Retrieval poisoning	Malicious content is inserted or boosted	Ingestion access; weak review/provenance	Persistent output manipulation	OWASP LLM04/08; NIST AML poisoning
Indirect prompt injection	Retrieved text carries adversarial instruction	RAG/agent trusts untrusted content	Tool abuse, disclosure, policy bypass	OWASP LLM01; MITRE ATLAS prompt-injection family
Metadata poisoning	Security or ranking attributes are falsified	Contributor can set tenant/classification/trust fields	Authorization failure, rank manipulation	Data-integrity and poisoning relationship
Embedding-model drift	Model/version changes without revalidation	Mutable pipeline or provider	Changed neighbours, leakage, degraded grounding	NIST AI RMF measurement/change risk
Orphaned vectors	Source is deleted or reclassified but vectors persist	Weak deletion propagation	Retention breach, stale decisions	Privacy, lifecycle governance
Query inference	Repeated queries reveal membership or semantic properties	Search access; rich scores/results	Sensitive information inference	NIST AML privacy attacks; OWASP LLM08
Raw export exposure	Embeddings, metadata or source chunks are exported broadly	Excess privilege or insecure backup	Bulk disclosure, offline inference	OWASP LLM02/08
Index exhaustion	Adversary floods vectors or expensive searches	Unbounded ingestion/query	Cost, latency, outage	OWASP LLM10
Hybrid-search mismatch	Vector branch is filtered but keyword/sparse branch is not	Multiple retrieval paths	Authorization bypass	OWASP LLM08; access control
Backup/replica gap	Secondary copies lack production controls or deletion	Weak lifecycle design	Persistent disclosure, residency breach	NIST CSF data security

3c. Impact analysis

Ratings are checklist priorities, not verified UAIF™ severity identifiers.

Impact	Example	Priority	Affected stakeholders
Technical	Poisoned chunks dominate retrieval and redirect an agent	High–critical	Engineering, security, users
Confidentiality	Query returns another tenant's protected document	Critical	Customer, privacy, legal
Integrity	Falsified metadata marks an untrusted source as approved	High	Data owner, business owner
Availability	Vector flooding or pathological search exhausts capacity	High	Operations, customers
Financial	Wrong retrieved policy drives unauthorized payment or advice	High–critical	Finance, customers, board
Regulatory	Deleted personal data remains searchable in vectors/backups	High	DPO, legal, affected people
Reputational	Internal confidential content appears in a public response	High	Communications, leadership
Safety	Retrieval suppresses current safety procedure and returns obsolete instruction	Critical	Safety owner, end users

3d. Documented findings and evidence

Source / finding	Evidence	What it establishes	Practitioner lesson	Notably absent
NIST AI 100-2e2025 adversarial ML taxonomy	**T1:** NIST publication	Poisoning and privacy attacks are lifecycle security concerns; RAG introduces external resources into the generative system boundary.	Treat source, embedding, retrieval and downstream model as one attack surface.	It does not prescribe product-specific vector-database configuration or guarantee a mitigation.
NIST AI 600-1 Generative AI Profile	**T1:** NIST publication	GAI value chains and RAG data require provenance, measurement, security testing and reassessment.	Reassess after source, embedding, chunking, index or retrieval-policy changes.	It is voluntary risk-management guidance, not a product certification or legal determination.
OWASP LLM08:2025 Vector and Embedding Weaknesses	**T1:** official OWASP project guidance	Weak generation, storage and retrieval of vectors/embeddings can enable poisoning, manipulation and sensitive-data exposure.	Enforce permission-aware partitioning, provenance, validation and monitoring across the lifecycle.	OWASP classification does not establish prevalence or prove exploitation of a particular deployment.
CVE-2026-45829 — ChromaDB Python server	**T1:** CVE/NVD record	A pre-authentication code-injection condition allowed attacker-controlled model configuration with `trust_remote_code` to be processed through a collection-creation endpoint in affected versions.	Keep vector services off untrusted networks, authenticate before processing configuration, prohibit unapproved remote code/model loading and confine the workload.	The record establishes the vulnerable condition, not exploitation in a particular ODA3 or reader environment; current affected/fixed versions must be checked against vendor guidance.
CVE-2026-45830 — ChromaDB authorization validation	**T1:** CVE/NVD record	A lack of authorization validation allowed an authenticated user to access or modify collections across tenants in affected versions.	Treat product tenant constructs as untrusted until authorization is negatively tested; add compensating gateway controls where required.	The record does not establish that every deployment or vector product has the same defect, or that namespace isolation is universally ineffective.
CVE-2025-64513 — Milvus Proxy authentication bypass	**T1:** vendor security advisory and NVD record	An unauthenticated attacker could bypass authentication in affected Milvus Proxy versions and obtain administrative access to cluster data and operations.	Patch supported branches, restrict network reachability, test authentication at every exposed protocol/endpoint and validate temporary gateway mitigations independently.	The record does not establish exploitation in a particular deployment; affected/fixed versions and mitigations must be checked against the current vendor advisory.
CVE-2026-41705 — Spring AI MilvusVectorStore expression injection	**T1:** Spring security advisory and NVD record	Unsanitized document IDs in the Spring AI Milvus integration could alter a filter expression used by a delete operation.	Include application frameworks, SDKs and query builders in the vector security boundary; use typed/parameterized expressions and	This vulnerability is attributed to the Spring AI integration layer, not Milvus core, and does not establish that every filter implementation is

			adversarial deletion tests.	affected.
Zou, Geng, Wang & Jia — "PoisonedRAG" (arXiv 2402.07867, 2024; USENIX Security 2025)	**T1:** peer-reviewed academic publication	Injecting a small number of crafted texts into a RAG knowledge database can reliably induce an attacker-chosen answer to an attacker-chosen target question — the first systematic knowledge-corruption attack against RAG.	Retrieval integrity is a security property independent of model-level guardrails; a RAG system trusting retrieved context without verifying provenance is exploitable even with a fully safe model.	The study is an academic demonstration; it does not establish a confirmed production exploitation rate or that every RAG deployment is equally vulnerable.

4. Technical and Structural Deep Dive

4.1 Secure lifecycle architecture

```
Authoritative sources
  ownership • classification • licence • retention • tenant
              ↓ controlled acquisition
Ingestion trust boundary
  malware/parser checks → content review → chunking → embedding
  provenance + policy metadata + version manifest
              ↓ approved write identity
Vector store
  tenant isolation • encryption • backups • audit • quotas
              ↓ authenticated query
Policy enforcement point
  identity → current entitlement → server-side filter → bounded search
              ↓ candidates
Post-retrieval controls
  provenance check • rerank • instruction isolation • output/action policy
              ↓
Model / agent / human decision + complete retrieval trace
```

4.2 Security mechanics

1. **Register the use case:** record owner, purpose, users, tenants, data classes, downstream decisions and prohibited uses.
2. **Approve sources:** authenticate origin; confirm ownership, licence, classification, residency, retention and refresh authority.
3. **Minimize before embedding:** detect credentials, secrets, unnecessary personal data and out-of-scope content; reject, redact or tokenize according to approved policy before material reaches the embedder.
4. **Transform deterministically:** version parser, normalization, chunker and embedding model; retain hashes and lineage.
5. **Write through a dedicated identity:** separate ingestion, query, administration, backup and deletion privileges.
6. **Bind security metadata:** derive tenant and access attributes from authoritative systems; do not accept caller-supplied labels as truth.
7. **Authorize every query:** resolve user/service identity and current entitlement before search; enforce filters server-side and fail closed.
8. **Limit disclosure:** return the minimum chunks and metadata required; suppress raw scores, internal IDs or vectors unless justified.

9. **Treat retrieved content as untrusted:** distinguish data from instruction; prevent retrieved text from granting tools or overriding policy.
10. **Trace downstream use:** join candidates, returned chunks, model response, human approval and action.
11. **Reconcile lifecycle events:** reclassification, revocation, deletion and retention changes must propagate to indexes, caches, replicas and backups.

4.3 Enabling-condition diagnostics

Condition	Diagnostic test
Namespace assumed to be isolation	Can an adversarial query cross namespaces through API, alias, backup or hybrid search?
Model-authored filters	Can the LLM remove or broaden an access predicate?
Client-only authorization	Does the database accept a query without the trusted policy filter?
Caller-supplied tenant metadata	Can an ingestion client label data as another tenant or approved source?
Unversioned embeddings	Can operations identify the model, dimensions and preprocessing for every vector?
Stale entitlements	How quickly does revoked access disappear from retrieval results and caches?
Incomplete deletion	Can a deleted source still be found in index, replica, snapshot or evaluation cache?
Score disclosure	Do exact similarity scores enable systematic probing?
Shared administrator	Can one account ingest, query, export, delete and disable audit?
No negative testing	Has cross-tenant retrieval been tested with adversarial filters and hybrid search?

4.4 Observable indicators

Stage	Indicators
Reconnaissance	High-volume semantically related queries; systematic threshold probing; unusual score requests
Ingestion abuse	New source/domain, abnormal contributor, duplicate clusters, hidden instruction patterns, metadata mismatch
Poisoning/persistence	Sudden rank dominance, content with high similarity across unrelated queries, unapproved index alias
Authorization failure	Returned tenant differs from requester; empty/malformed filter; fallback from filtered to global search
Exfiltration	High result count, raw vector export, unusual metadata fields, new destination or bulk pagination
Impairment	Audit disabled, index replaced, retention changed, tombstones removed, backups restored unexpectedly

4.5 Vulnerable versus hardened pattern

Vulnerable baseline	Hardened pattern
One shared collection and client-supplied tenant filter	Tested isolation plus policy-derived, server-enforced tenant predicate
Any uploader writes chunks and security metadata	Staged ingestion with separate approval and derived metadata
“Vector is not personal data” assumption	Classify derived representation from source, use and inference risk
Update embedding model in place	Build new versioned index, run security/regression tests, then

	controlled cutover
Delete source only	Trace and verify deletion across chunks, vectors, cache, replica, export and backup schedule
Log query text only	Log identity, policy decision, index, candidates, returned chunks and downstream use
Trust similarity score as confidence	Calibrate per model/use case and require provenance/authority checks

4.6 Reference query envelope

The application should construct an immutable authorization envelope outside the model. Product syntax varies.

```
{
  "request_id": "ret-2026-0042",
  "principal": "user:78421",
  "purpose": "support_case_response",
  "index_version": "support-kb@2026-07-17",
  "embedding_model": "approved-embedder@immutable-version",
  "mandatory_filter": {
    "tenant_id": "tenant-018",
    "classification": {"$in": ["public", "tenant-internal"]},
    "document_state": "active",
    "entitlement_epoch": 1842
  },
  "retrieval_limits": {"top_k": 8, "max_chunks_per_source": 2},
  "return_fields": ["chunk_ref", "source_ref", "provenance_ref"],
  "deny_on_filter_error": true
}
```

Security properties: the trusted service supplies the principal and mandatory filter; the model may suggest search terms but cannot delete policy predicates; errors deny rather than fall back to unfiltered search.

5. Practitioner Decision Framework

5a. Risk scoring aid

Score 1–4, multiply by weight and divide by 100. This prioritizes work; it is not a UAIF™ severity score or conformity method.

Factor	Weight	1 — Low	2 — Moderate	3 — High	4 — Critical
Source sensitivity	20	Public	Internal	Confidential/personal	Secrets, special-category, safety-critical
Tenant exposure	15	Dedicated/ isolated	Shared infrastructure, hard isolation	Shared collection with policy filter	Client/model filter or unknown isolation
Downstream consequence	20	Search assistance	Draft recommendation	Material human decision	Autonomous/irreversible action
Ingestion trust	15	Curated, signed	Controlled contributors	External sources with review	Open/untrusted automated ingestion
Authorization strength	15	Server-side, tested	Server-side, limited tests	Mixed/client-dependent	Missing or fail-open
Provenance/ deletion	10	Complete and verified	Mostly traceable	Partial/stale	Untraceable
Observability	5	Full retrieval trace	Strong audit	Partial candidates/logs	No reconstructable trace

Suggested bands: 1.00–1.49 Low; 1.50–2.24 Moderate; 2.25–2.99 High; 3.00–4.00 Critical. Validate against the organization's risk appetite.

Hard stops: cross-tenant result; active poisoning of consequential workflow; exposed administrative endpoint; unknown owner for sensitive collection; untraceable deletion failure; model-generated authorization predicate; material retrieved instruction capable of invoking a privileged tool.

5b. Decision tree

```
Does the store contain or derive from sensitive, tenant-separated or safety-relevant data?
  YES → Require registered owner, classification, server-side authorization,
        provenance, tested deletion and full retrieval trace before production.
  NO  → Apply baseline controls and document classification rationale.

Can a model, browser or untrusted client alter mandatory authorization filters?
  YES → Stop production use; move policy construction to a trusted enforcement service.
  NO  → Test fail-closed behaviour and every hybrid-search branch.

Can external or user-controlled content enter the index?
  YES → Stage ingestion, scan and classify content, derive security metadata,
        approve promotion and provide an independent ingestion kill switch.
  NO  → Verify source integrity and change authorization.

Has model, chunking, metric, reranker, index or filter policy changed?
  YES → Create a new versioned bundle; run leakage, authorization, poisoning
        and retrieval-quality regression tests before cutover.

Did an unauthorized or poisoned result reach a user/model/agent?
  YES → Create a UAIF™-structured incident record; preserve trace and index state;
        contain through the applicable AI-IRF™ path; assess downstream decisions.
```

5c. Prioritization

Scenario	Risk	Recommended action	Owner	Target
Cross-tenant retrieval reproduced	Critical	Disable affected query path; preserve evidence; inspect all related traces	Incident commander / platform	Immediate
Publicly reachable admin or export API	Critical	Restrict network/IAM; rotate credentials; review access	Security/platform	Immediate
Model controls tenant predicate	Critical	Remove model authority; enforce trusted server-side policy	Application security	Before further use
Untrusted feed can auto-publish	High	Suspend ingestion; stage and approve promotion	Data/RAG owner	Same operational period
Embedding model changed without evaluation	High	Freeze cutover; build versioned regression results	MLOps	Before release
Source deleted but vectors remain	High	Quarantine results; complete lifecycle deletion and verify	Data/privacy owner	Risk/legal based
No evidence of negative authorization tests	Moderate–high	Build cross-tenant and malformed-filter suite	Independent tester	Before material scale-up

6. Security Controls Matrix

Mappings are functional relationships, not equivalence, compliance, certification or endorsement. ISO clause-level use requires licensed standards text.

Control	Requirement	GAISSF™	UAIF™	AI-IRF™	Evidence	External alignment
VDB-01 Registry and ownership	Register collection, purpose, owner, tenants, sources, region, downstream use and criticality.	Governance/ accountability	System context	Preparation	Reconciled register	NIST AI RMF GOVERN/MAP; ISO 42001 governance
VDB-02 Source approval	Authenticate and approve source, licence, classification, retention and refresh authority.	Data control	Causality context	Identification/ eradication	Source manifest	NIST AI 600-1; OWASP LLM03/04
VDB-03 Segregated ingestion	Separate acquire, transform, approve and production-write privileges.	Access/ change control	Affected identity	Containment	IAM and workflow tests	ISO 27001 access/change; OWASP LLM04
VDB-04 Derived security metadata	Create tenant/classification fields from authoritative policy, not caller claims.	Authorization objective	Boundary/ evidence	Containment	Mapping and negative tests	OWASP LLM08; NIST CSF PR.AA
VDB-05 Query-time authorization	Authenticate principal and enforce current server-side predicates for every search branch.	Access control	Exposure scope	Containment	Policy decision trace	OWASP LLM08; ISO 27001 access control
VDB-06 Tenant isolation	Use tested logical/physical isolation proportionate to impact.	Segregation objective	Affected tenants	Containment	Cross-tenant tests	OWASP LLM08; NIST CSF PR.DS
VDB-07 Version-pinned bundle	Bind embedder, parser, chunker, metric, index, reranker and policy.	Approved boundary	Component identity	Recovery	Signed manifest	NIST AI RMF MEASURE/MANAGE
VDB-08 Provenance and lineage	Trace source through transformations, vector, chunk, index and returned result.	Evidence expectation	Evidence/ causality	All phases	Lineage samples	NIST AI 600-1; EU AI Act data/logging context where applicable
VDB-09 Encryption and key separation	Encrypt transport/storage/backups; separate key administration and rotation.	Data protection	Impact context	Containment/ recovery	Config and rotation test	NIST CSF PR.DS; ISO 27001 cryptography
VDB-10 Least-privilege APIs	Separate query, ingest, admin, export, backup and delete identities.	IAM objective	Affected identity	Containment	Entitlement review	NIST CSF PR.AA; ISO 27001 access control
VDB-11 Retrieval minimization	Limit top-k, fields, chunks/source, raw score/vector exposure and pagination.	Minimization control	Exposure measurement	Containment	Config and query test	OWASP LLM02/08/10
VDB-12 Poisoning defenses	Stage content, scan, deduplicate, validate metadata and detect rank anomalies.	Integrity control	Incident classification	Identification/ eradication	Promotion records, alerts	NIST AML; OWASP LLM04/08
VDB-13 Instruction isolation	Treat retrieved content as untrusted data; constrain tools and authorization independently.	System safety/control	Attack-path evidence	Containment	Agent/prompt tests	OWASP LLM01/06; MITRE ATLAS
VDB-14 Lifecycle deletion	Propagate deletion/reclassification to chunks, vectors, cache, replicas and	Lifecycle/privacy control	Impact/ closure	Eradication/ recovery	Deletion proof	NIST CSF PR.DS; privacy law as applicable

	backups.					
VDB-15 Audit and retrieval trace	Join identity, policy, candidates, results, versions and downstream action.	Evidence expectation	Incident evidence	Identification	Reconstructable sample	NIST CSF DE/RS; ISO 27001 logging
VDB-16 Quotas and abuse limits	Bound ingestion, query rate, top-k, dimensions, batch/export and compute.	Resilience control	Impact	Containment	Load/abuse tests	OWASP LLM10; NIST CSF PR.IR
VDB-17 Independent ingestion stop	Suspend source refresh and production promotion without relying on consuming app.	Resilience/ response	Containment record	Containment	Kill-switch drill	OWASP LLM01/04/08
VDB-18 Backup/restore assurance	Apply isolation, encryption, residency, retention and deletion to secondary copies; test restore.	Resilience/ evidence	Scope evidence	Recovery	Restore test	NIST CSF RC; ISO 27001 backup
VDB-19 Security regression suite	Test authorization, leakage, poisoning, drift and deletion before change/cutover.	Assurance	Evidence/ classification	Recovery/ lessons	Raw results	NIST AI RMF MEASURE/MANAGE
VDB-20 Supplier evidence	Contract for notice, version/change history, logs, preservation, deletion and region.	Supplier governance	External evidence	Joint response	Contract/exercise	NIST CSF GV.SC; ISO supplier controls
VDB-21 Vulnerability and patch management	Inventory product, package, image and plugin versions; monitor advisories/CVEs; risk-assess and patch or compensate within defined time.	Technology-risk control	Component and exposure evidence	Preparation/ eradication	SBOM, scan, patch/exception record	NIST CSF ID.RA/PR.PS; ISO 27001 vulnerability management
VDB-22 Workload and code-loading confinement	Disable unapproved remote code/model loading; run the service with non-root identity, minimal filesystem/secrets, constrained egress and bounded execution.	Platform /security objective	Execution and impact context	Containment/ recovery	Runtime policy and escape/egress test	NIST CSF PR.PS/PR.IR; OWASP LLM03
VDB-23 Private network and mutual service authentication	Remove direct public reachability; restrict ingress/egress through private endpoints, allow-lists and authenticated service channels such as mTLS where supported.	Network /trust objective	Access path and exposure	Preparation/ containment	Reachability scan, firewall and certificate test	NIST CSF PR.IR/PR.AA; ISO 27001 network security
VDB-24 Pre-embedding minimization and secret detection	Detect and remove or tokenize prohibited credentials, secrets, unnecessary personal data and out-of-scope content before chunking/embedding; record approved exceptions.	Data minimization/control	Exposed-data evidence	Preparation/ eradication	Scan policy, findings, redaction and exception samples	NIST AI 600-1; NIST CSF PR.DS; OWASP LLM02/04

7. Detection & Monitoring

7a. Logging requirements

Source	Minimum fields	Purpose	Priority
Query gateway	request/trace ID, principal, purpose, tenant, filter hash, policy result, top-k, time	Authorization and scoping	P1
Vector store	collection/index/version, candidates/returned refs, scores or protected score class, latency, result count	Reproduction and anomaly detection	P1
Ingestion	source, actor, hashes, parser/chunker/embedder versions, metadata derivation, approval, write set	Poisoning and provenance	P1
IAM/control plane	grants, admin actions, export, key, alias/index and audit changes	Privilege and impairment	P1
Deletion lifecycle	source request, affected object/vector/cache/replica status, backup schedule, verifier	Retention and privacy proof	P1
Downstream AI	retrieved references, prompt/context reference, output, tool authorization and action	Impact reconstruction	P1

Priority labels are operational logging priorities, not UAIF™ severity tiers.

7b. Detection logic

```
Rule: Tenant Boundary Mismatch
Source: Query gateway + returned metadata
Logic: any(returned.tenant_id != authorized_tenant_id)
Threshold: one event
Action: block response, preserve trace, suspend affected path, open incident
```

```
Rule: Missing Mandatory Filter
Source: Policy enforcement + vector query audit
Logic: policy_required = true AND
       (filter_hash is null OR policy_decision != "allow") AND query_executed = true
Threshold: one event
Action: fail closed and escalate
```

```
Rule: Retrieval Rank Dominance
Source: Retrieval telemetry
Logic: same_source_share(top_k) > approved_limit across diverse queries
       OR source_rank_shift > validated_baseline
Threshold: use-case calibrated
False positives: authoritative high-use source, scheduled index rebuild
```

```
Rule: Orphaned Vector
Source: Source registry + index inventory
Logic: vector.source_state IN (deleted, expired, access_revoked)
       AND vector.searchable = true
Threshold: one sensitive object or defined batch tolerance
```

```
Rule: Enumeration / Inference Pattern
Source: Query telemetry
Logic: high semantic-neighbour query volume + threshold stepping + low result diversity
Threshold: identity/use-case baseline
Action: throttle, reduce disclosure, investigate intent
```

7c. Indicators of control weakness

Indicator	Confidence	Interpretation
Query succeeds when authorization service is unavailable	High	Fail-open retrieval path
Returned object lacks source/provenance reference	High	Unreconstructable retrieval
Same API key performs ingest, query, export and delete	High	Excess privilege and weak attribution
Index alias changes outside release workflow	High	Unauthorized or unassured cutover
Collection/index request references an unapproved model repository or enables remote-code trust	High	Code-loading or supply-chain attack candidate
Vector service accepts configuration-changing requests before authentication	High	Pre-authentication attack surface
Vector protocol, proxy, debug or management endpoint is reachable outside the approved private boundary	High	Authentication-bypass and administrative exposure candidate
Delete or filter request contains unescaped operators derived from an external identifier	High	Integration-layer expression-injection candidate
Deleted source remains in regression/search result	High	Lifecycle propagation failure
Similarity scores or raw vectors exposed to untrusted clients	Medium–high	Inference/probing surface
New external source dominates unrelated queries	Medium–high	Poisoning or pipeline failure candidate

7d. Response health metrics

Metric	Definition	Decision use
Authorization negative-test pass rate	Denied cross-tenant/malformed-filter tests ÷ total negative tests	Release gate
Provenance completeness	Returned chunks with complete source and transformation chain	Evidence readiness
Deletion verification time	Request to confirmed non-searchability across active copies	Privacy/lifecycle risk
Unauthorized-result rate	Blocked or discovered results outside policy boundary	Incident trend
Index drift rate	Unapproved differences between manifest and runtime	Change/supplier risk
Time to ingestion suspension	Trigger to verified stop of production refresh	Containment readiness

8. Incident Response and Escalation

Phase	Actions	Owner	Evidence	Decision point
Preparation	Inventory stores; test authorization; prepare snapshots, ingestion stop, key rotation and rollback.	Governance/ platform/IR	Register, tests, authority	Can each consequential store be contained independently?
Identification	Create incident ID; preserve query/retrieval trace, index manifest, source and IAM state; classify impact and causality provisionally.	IR/SOC	UAIF™-structured record	Cross-tenant, poisoning, privacy or safety escalation?
Containment	Block response; suspend affected query and ingestion paths; revoke	Incident	Before/after	Scoped restriction or full store

	identities; quarantine content/index without deleting evidence.	commander	state	shutdown?
Eradication	Remove compromised source/identity/config; rebuild clean versioned index; search portfolio for same condition.	Data/ platform/ security	Rebuild manifest, hunt	Root condition removed across all copies?
Recovery	Test tenant isolation, authorized recall, negative leakage, provenance and downstream behaviour; stage cutover.	Independent tester/owner	Raw test results	Recovery criteria met on exact bundle?
Lessons	Finalize causality and impact; update GAISSF™ controls; retest; link related incidents through UAIF™ function.	Governance/ assurance	Corrective action and retest	Close, extend or reopen?

Code-execution escalation: If exploitation or attempted exploitation of a vector-service code-execution vulnerability is plausible, isolate the workload and treat its host identity, mounted secrets, tokens, connected stores and downstream credentials as potentially compromised. Preserve volatile evidence where safe, rotate/revoke exposed credentials, and rebuild from a verified image rather than assuming that patching the database alone eradicates persistence.

Vector-specific evidence package

12. Collection/index/alias and immutable version manifest.
13. Embedding model, dimensions, preprocessing, chunking, metric and reranker versions.
14. Source objects or protected references, hashes, ownership and classification.
15. Ingestion actor, transformations, metadata derivation and approval.
16. Query identity, mandatory policy, candidates, returned chunks and scores/class.
17. Downstream prompt/output, human decision, tool call and action.
18. IAM, control-plane, export, deletion, replica and backup events.
19. Snapshot plus evidence manifest and custody record.

Do not rebuild or compact the affected index before preserving the state needed to reproduce ranking, authorization and provenance. Protect people and stop active disclosure first when preservation would prolong harm.

9. Audit & Assurance Checklist

9a. Documentation

Artefact	Required content	Owner	Review
Vector store register	Purpose, owner, tenants, region, sources, downstream use	AI governance	Quarterly/ change
Data-flow/boundary diagram	Source through ingestion, store, retrieval and action	Architect	Material change
Authorization design	Identity, policy source, enforcement, failure behaviour	IAM/app security	Semiannual
Retrieval bundle manifest	All versioned components and approvals	MLOps/ platform	Every release
Retention/deletion specification	Source, vector, cache, replica, backup obligations	Data/privacy	Annual/ change
Incident runbook	Query block, ingestion stop, snapshot, rebuild, notification	IR lead	Exercise/ incident

9b. Technical evidence

Control area	Test	Expected evidence
Tenant isolation	Attempt cross-tenant query through every API and hybrid branch	Denial plus audit event
Fail closed	Disable or corrupt authorization dependency	No unfiltered query executes
Metadata integrity	Submit forged tenant/classification at ingestion	Rejection or authoritative overwrite
Poisoning resistance	Insert adversarial/duplicate/high-similarity content in staging	Detection, quarantine and bounded rank
Deletion	Delete/reclassify source and search all active copies	Non-searchability plus tracked backup expiry
Rollback	Restore previous complete retrieval bundle	Verified manifest and regression pass
Evidence	Reconstruct sampled answer/action from trace	Exact source and policy chain

9c. Assurance interview questions

20. Show who owns each collection and who can suspend ingestion outside business hours.
21. Demonstrate that an LLM cannot remove the mandatory tenant predicate.
22. What happens if the authorization service times out?
23. Trace one returned chunk to its source, transformations, approval and deletion state.
24. Show a negative cross-tenant test for vector, keyword and hybrid search.
25. How do you detect a new document dominating semantically unrelated queries?
26. Prove that the active embedding model and index match the approved manifest.
27. Where do deleted vectors remain, for how long, and under whose approval?
28. Can administrators export raw vectors without independent approval and audit?
29. Show how a vector-related incident changed a control and was independently retested.

9d. Maturity

Level	Observable state
1 — Ad hoc	Unregistered stores, shared keys, weak provenance, client filters
2 — Defined	Owners, policies and baseline encryption documented; tests inconsistent
3 — Operational	Server-side authorization, version manifests, traces, deletion and drills operate
4 — Measured	Leakage, provenance, deletion, drift and containment metrics drive decisions
5 — Adaptive	Threat-informed tests and incidents change controls across the portfolio

10. Research Insight ★

Finding — security boundary, not search accessory: NIST's adversarial ML taxonomy models RAG as a generative-AI system that consumes external resources, often through a vector database. OWASP separately identifies vector and embedding weaknesses as a material LLM application risk. **Evidence: T1 within each publication's scope.**

Technical significance: Security properties do not reside in the vector store alone. Source trust, embedding transformation, authorization metadata, retrieval policy and downstream instruction handling jointly determine exposure. A well-configured database can still return unauthorized or poisoned context if the application constructs unsafe predicates or the ingestion pipeline supplies false metadata.

Operational lesson: Test one complete retrieval decision—not only database settings. Begin with the

human or service identity, reproduce the policy decision, inspect candidate and returned chunks, verify provenance, and follow the context into the model and any downstream action.

Notably Absent: These sources do not establish a universal safe similarity threshold, a single preferred isolation architecture or quantified control effectiveness across products.

11. Common Mistakes

Mistake	Corrective approach
Treating a namespace as proven isolation	Test enforcement through every data, admin, alias, backup and hybrid path.
Allowing the LLM to create security filters	Construct mandatory policy in a trusted service and prevent removal.
Trusting caller-supplied tenant metadata	Derive security attributes from authoritative identity and data systems.
Classifying embeddings as non-sensitive by default	Base controls on source, use, inference and linkage risk.
Rolling an embedder in place	Version the full bundle; rebuild and regress before cutover.
Deleting only the original document	Verify chunks, vectors, caches, replicas, exports and backup lifecycle.
Logging only the final answer	Preserve identity, policy, candidates, returned sources and downstream action.
Treating high similarity as truth	Check source authority, freshness, contradictions and use-case calibration.
Scanning documents but not metadata	Validate tenant, classification, source, timestamps, boosts and state.
Sharing one powerful API key	Separate query, ingestion, admin, export, backup and deletion identities.
Containing the chat UI only	Suspend affected ingestion, refresh, index and downstream action paths.
Restoring when search “looks normal”	Require security, retrieval-quality, safety and negative authorization gates.

12. Quick Reference

12.1 Security checklist

#	Check	Yes	No	Partial
1	Every store, collection and index has an owner, purpose and data classification.	☐	☐	☐
2	Source ownership, licence, retention, residency and refresh authority are approved.	☐	☐	☐
3	Ingestion, approval, query, administration, export and deletion identities are separated.	☐	☐	☐
4	Tenant/classification metadata is derived from authoritative policy.	☐	☐	☐
5	Query authorization is server-side, current and fail-closed.	☐	☐	☐
6	Vector, keyword and hybrid paths enforce identical mandatory policy.	☐	☐	☐
7	Cross-tenant and malformed-filter negative tests pass.	☐	☐	☐
8	Embedding model, preprocessing, chunking, index, metric and reranker are version-bound.	☐	☐	☐
9	Every returned chunk has reconstructable provenance.	☐	☐	☐

10	Retrieved content is treated as untrusted and cannot grant tool authority.	☐	☐	☐
11	Top-k, fields, pagination, exports and raw vector/score exposure are minimized.	☐	☐	☐
12	Poisoning, duplicate/rank dominance and abnormal queries are monitored.	☐	☐	☐
13	Deletion/reclassification propagates to active copies and tracked backups.	☐	☐	☐
14	Independent query and ingestion suspension mechanisms are tested.	☐	☐	☐
15	Product versions, CVEs and patches are tracked; remote code/model loading is disabled or tightly allow-listed and confined.	☐	☐	☐
16	Data, proxy, debug and management interfaces are private and mutually authenticated where supported.	☐	☐	☐
17	Secrets, credentials and unnecessary personal data are detected and minimized before embedding.	☐	☐	☐
18	Retrieval incidents feed UAIF™ records, AI-IRF™ response and GAISSF™ improvement.	☐	☐	☐

12.2 Risk-to-control

Risk	Primary controls
Cross-tenant leakage	VDB-04, VDB-05, VDB-06, VDB-19
Retrieval poisoning	VDB-02, VDB-03, VDB-08, VDB-12, VDB-17
Indirect prompt injection	VDB-12, VDB-13, VDB-15, VDB-17
Embedding/index drift	VDB-07, VDB-19
Sensitive inference/export	VDB-09, VDB-10, VDB-11, VDB-16
Orphaned vectors	VDB-08, VDB-14, VDB-18
Unreconstructable incident	VDB-07, VDB-08, VDB-15
Supplier evidence gap	VDB-20
Product vulnerability / code execution	VDB-10, VDB-16, VDB-21, VDB-22
Authentication bypass / exposed service	VDB-05, VDB-10, VDB-21, VDB-23
Integration-layer expression injection	VDB-05, VDB-19, VDB-21
Credential or unnecessary personal-data capture	VDB-02, VDB-08, VDB-11, VDB-14, VDB-24

12.3 Release gates

Gate	Required result
Scope	Owner, tenants, data, downstream decisions and prohibited uses approved
Authorization	All negative tests deny and generate evidence
Provenance	Sampled returned chunks trace end to end
Poisoning	Staged adversarial sources are detected/contained within approved tolerance
Privacy	Minimization, retention, deletion and residency decisions approved
Regression	Retrieval quality and security tests pass on exact bundle
Response	Query/ingestion kill paths and clean rollback exercised
Residual risk	Authorized owner accepts bounded, time-limited exceptions

13. Assessment and Certification Readiness

This section addresses readiness only. It does not state that ODA3 certification, accreditation, licensed assessment delivery or regulatory recognition is operational.

Framew	Assessor focus	Common weakness	Evidence	Mature indicator

ork / standard				
ODA3 suite	GAISSF™ controls connect to UAIF™ incident evidence, AI-IRF™ response and verified improvement	Framework names present but retrieval evidence is disconnected	Complete control-to-incident-to-retest trace	One reproducible vector incident loop
NIST AI RMF / AI 600-1	Governed data/value chain, measured risks, provenance and change reassessment	RAG added without new measurement	Register, evaluation, lineage and risk treatment	Retrieval changes trigger reassessment
NIST CSF 2.0	Governance, identity, data security, detection, response and recovery	Vector store absent from asset/data scope	Profile, telemetry, incidents, restoration	Unified cyber and AI controls
ISO/IEC 42001:2023	AIMS scope, data governance, operation, evaluation and improvement	Vector pipeline outside AIMS evidence	Scope, risks, controls, monitoring, corrective action	Lifecycle evidence supports management review
ISO/IEC 27001:2022	Access, cryptography, logging, supplier, backup and incident controls	Generic DB policy without AI retrieval test	IAM, encryption, logs, supplier and restore tests	Technical controls tested end to end
OWASP LLM Top 10 2025	Prompt injection, disclosure, poisoning, vector/embedding and consumption risks	Checklist labels without abuse cases	Threat model and adversarial tests	Tests cover ingestion through downstream action
EU AI Act	Applicable role/classification, data governance, logs, robustness, monitoring and incident duties	Generic mapping without applicability decision	Counsel-approved scope and operating evidence	Duties embedded in routing where applicable

Examiner traps

30. **Namespace theatre:** a product setting is accepted as tenant isolation without adversarial cross-tenant testing.
31. **Encryption theatre:** data is encrypted but a shared application identity can retrieve every tenant.
32. **Provenance theatre:** source URL exists, but transformation, approval, served version and deletion state cannot be reconstructed.
33. **Deletion theatre:** a source is removed from the application while vectors remain searchable or recoverable indefinitely.
34. **Regression theatre:** recall metrics pass, but negative authorization and poisoning cases were never run.

Tier 1 versus Tier 2 assessment

Dimension	Tier 1 — documentation review	Tier 2 — operating/technical test
Inventory	Inspect register and diagrams	Reconcile runtime indexes and aliases
Authorization	Review policy and architecture	Execute cross-tenant, fail-closed and hybrid tests
Provenance	Inspect schema	Trace live returned chunks to source
Poisoning	Review promotion controls	Insert controlled adversarial content in staging
Deletion	Review procedure	Delete/reclassify and search all active copies
Response	Review runbook	Suspend ingestion/query, snapshot and restore clean bundle

Improvement	Inspect corrective register	Reperform control changed by incident/exercise

Preparation timeline

Period	Objective
Weeks 1–2	Inventory stores, owners, data classes, tenants, sources and critical retrieval paths.
Weeks 3–5	Correct identities, policy enforcement, metadata derivation, encryption and logging.
Weeks 6–8	Establish provenance, deletion, version manifests, backups and supplier evidence.
Weeks 9–11	Build authorization, poisoning, leakage, drift and retrieval regression tests.
Weeks 12–14	Exercise containment/recovery, remediate findings and complete readiness review.

The timeline is illustrative; effort depends on portfolio size, product capabilities, data sensitivity, tenancy model and existing platform controls.

Primary References

- [NIST AI 100-2e2025 — Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations](https://csrc.nist.gov/pubs/ai/100/2/e2025/final)
- [NIST AI 600-1 — Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile](https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.600-1.pdf)
- [NIST AI Risk Management Framework 1.0](https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.100-1.pdf)
- [NIST Cybersecurity Framework 2.0](https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf)
- [OWASP LLM08:2025 — Vector and Embedding Weaknesses](https://genai.owasp.org/llmrisk/llm08-excessive-agency/)
- [OWASP LLM04:2025 — Data and Model Poisoning](https://genai.owasp.org/llmrisk/llm04-model-denial-of-service/)
- [OWASP LLM01:2025 — Prompt Injection](https://genai.owasp.org/llmrisk/llm01-prompt-injection/)
- [OWASP Top 10 for LLM Applications 2025](https://genai.owasp.org/llm-top-10/)
- [MITRE ATLAS](https://atlas.mitre.org/)
- [NVD — CVE-2026-45829: ChromaDB Pre-authentication Code Injection](https://nvd.nist.gov/vuln/detail/CVE-2026-45829)
- [NVD — CVE-2026-45830: ChromaDB Authorization Validation Failure](https://nvd.nist.gov/vuln/detail/CVE-2026-45830)
- [Milvus Security Advisory — CVE-2025-64513: Proxy Authentication Bypass](https://github.com/milvus-io/milvus/security/advisories/GHSA-mhjq-8c7m-3f7p)
- [NVD — CVE-2025-64513: Milvus Proxy Authentication Bypass](https://nvd.nist.gov/vuln/detail/CVE-2025-64513)
- [Spring Security Advisory — CVE-2026-41705: MilvusVectorStore Expression Injection](https://spring.io/security/cve-2026-41705)
- [NVD — CVE-2026-41705: Spring AI MilvusVectorStore Expression Injection](https://nvd.nist.gov/vuln/detail/CVE-2026-41705)
- [Regulation (EU) 2024/1689 — Artificial Intelligence Act](https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng)


```
Doc ID: ODA3-2026-07-CHT-SEC-012 (pending confirmation against file listing)
Topic: Vector Database Security Checklist
Version: 1.3
Classification: Public
Intended Audience: CISOs, AI Security Architects, Application Security Teams,
Platform Engineers, Data and MLOps Teams, Privacy and Legal Teams, Auditors
Evidence Confidence: T1 for primary NIST, OWASP and regulatory sources;
T2–T3 for operational synthesis and deployment-specific recommendations
```

```
Review Cycle: Quarterly recommended
Framework Cross-References: GAISSF™ v1.0; UAIF™ v1.0; AI-IRF™ v1.0;
NIST AI RMF 1.0; NIST AI 600-1; NIST AI 100-2e2025; NIST CSF 2.0;
ISO/IEC 42001:2023; ISO/IEC 27001:2022; OWASP LLM Top 10 2025; MITRE ATLAS;
EU AI Act where applicable
Known Evidence Gaps: Exact ODA3 internal identifiers were not verified from supplied
authoritative framework documents. Product-specific isolation, consistency, deletion,
backup and audit behaviours require deployment testing. External mappings are functional
and require licensed standards text for audit-grade clause determinations. PoisonedRAG
citation corrected from an earlier draft's "Chen et al." to the actual authors (Zou, Geng,
Wang, Jia).
Related Publications: CHT-SEC-009 (AI Security Evidence Collection Guide);
CHT-SEC-010 (AI Security Governance Model); CHT-SEC-011 (AI Security Incident Response
Playbook)
```