AI SECURITY ENGINEERING HANDBOOK · 2026

Chapter 09 · AI Supply Chain

Standalone study module for LMS delivery and required reading.

FORMAT	USE	SCOPE	AUDIENCE
Standalone PDF	Study module	Single chapter	Learners

CHAPTER 09

AI Supply Chain

HANDBOOK STUDY COMPANION: STUDY FRAME

Use this chapter to build vocabulary, judgment, and role-readiness. Pair it with the Field Guide when you need applied actions, checklists, and control execution.

STUDY FOCUS

STUDY FOCUS	WHY IT MATTERS
Model artifact integrity, dataset provenance, fine-tuning pipeline security, registry controls, adapters, and promotion gates.	AI supply chain risk spans code, packages, datasets, model weights, registries, providers, and serving platforms.

Study Outcomes

- Trace model artifacts from source to production use.
- Identify intake, integrity, license, registry, and rollback evidence.
- Reason about unsafe formats, public hubs, and adapter risk.

DOMAIN MAPPING

RELATED AIPSA DOMAINS	APPLIED NEXT STEP	WORKBENCH INSTRUMENTS	RELATED SERVICES
Model Supply Chain Security	Model supply chain security	Artifact Analyzer	AI Product Security Assessment

CERTIFICATION AND ASSESSMENT BOUNDARY

This chapter supports training, diagnostic preparation, scorecards, interviews, and role-readiness evaluation. It does not guarantee credential outcomes.

The organization that would never deploy a software dependency without reviewing its source, checking its hash, and verifying its license routinely deploys model weights downloaded from public hubs with none of those checks. The oversight is not negligence in most cases. It is a category error: the team that owns model deployment thinks in terms of performance and inference cost rather than supply-chain trust, and no one has set expectations otherwise. AI supply chain security exists to close that gap before an incident makes it visible.

The organization that would never deploy a software dependency without reviewing its source, checking its hash, and verifying its license routinely deploys model weights downloaded from public hubs with none of those checks.

HANDBOOK

A model artifact earns production eligibility by moving through a governed lifecycle. Each stage — intake review, integrity verification, license review, registry entry, promotion gate, and deployment — produces evidence. That evidence chain is what makes the supply chain auditable, reproducible, and defensible when a security question arises.

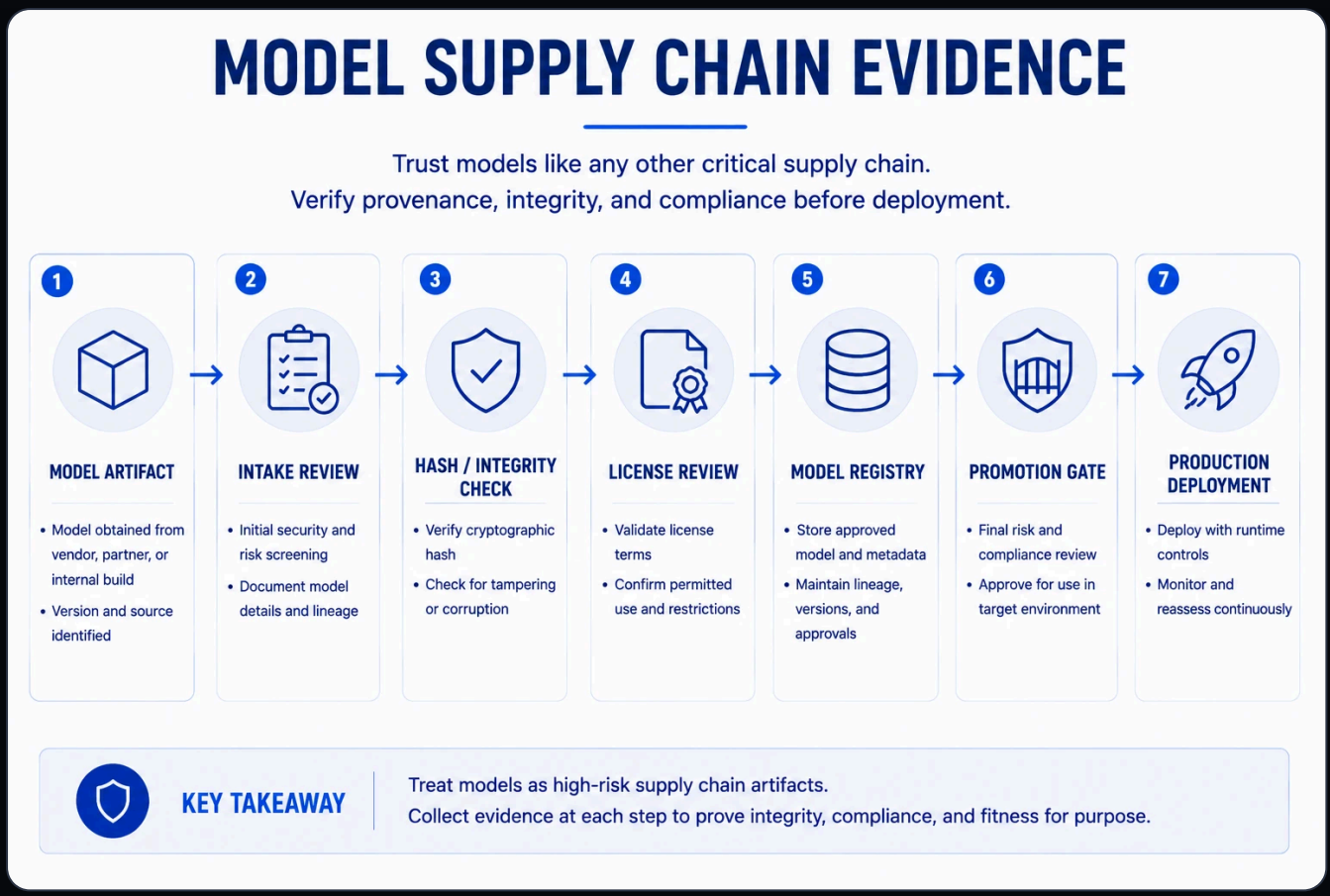

FIGURE 1: FIGURE 12: MODEL SUPPLY CHAIN EVIDENCE FLOW — ARTIFACT THROUGH INTAKE REVIEW, HASH/INTEGRITY CHECK, LICENSE REVIEW, MODEL REGISTRY, PROMOTION GATE, AND PRODUCTION DEPLOYMENT, WITH THE REGISTRY AND PROMOTION GATE AS THE GOVERNANCE ENFORCEMENT POINTS

CORE CONCEPTS

MODEL PROVENANCE

Provenance answers where the model came from, who created it, what it was trained or fine-tuned from, what license applies, and who approved it for production use. A complete provenance record identifies publisher, source URL, exact version, artifact hash, base model, fine-tuning process and data lineage where available, license terms, intended use, limitations, and named production owner. Provenance must be recorded before production promotion, not reconstructed when an incident requires it. A model without documented provenance cannot be reviewed, cannot be reproduced, and cannot be defended during a customer security review.

ARTIFACT INTEGRITY VERIFICATION

Integrity verification proves that the model artifact in production is the exact artifact that was approved. The core controls are: cryptographic hash verification before loading, immutable storage after review, deployment configuration pinned to a specific artifact hash, and registry promotion workflows that record the approving reviewer and the hash of the promoted artifact. Deploying by mutable references — "latest," a registry stage without a pinned hash, a branch reference — means the artifact can change without triggering re-approval. Integrity checks should run at promotion boundaries and at deployment, not only at download.

UNSAFE SERIALIZATION FORMATS

Some model and ML artifact formats execute code during loading. Pickle-based artifacts are the primary example in Python ML workflows: deserializing a pickle file can execute arbitrary code in the loading process's context, which in a model-serving environment often includes production credentials, object stores, and internal network access. Safer formats such as safetensors eliminate this risk for weight tensors. However, format safety is one control: it does not address provenance, license review, behavioral risk inherited from a base model, or tampering during distribution.

MODEL REGISTRY GOVERNANCE

A model registry becomes a security control only when it enforces metadata requirements, access control, versioning, approval gates, and promotion workflows rather than functioning as an organized file store. A production-eligible registry entry should include: owner, source, version, artifact hash, base lineage, license review outcome, eval evidence, approval record with reviewer identity and date, deployment targets, and rollback version. Promotion to production stages should require defined approvals that are visible and auditable. The registry is where supply chain evidence becomes operational.

LICENSE AND USE-RIGHTS

Model licenses can restrict commercial use, redistribution, derivative works, field of use, and output rights. Fine-tuning a base model may inherit the base model's license restrictions into the derived artifact. Deploying a model without license review creates legal and business risk that the security team may be asked to remediate after a product has shipped. License review must cover:

base model license, fine-tuning rights, training dataset terms, adapter licensing, and any restrictions on commercial deployment or customer-facing use.

THE PRACTITIONER'S CHALLENGE

The political challenge is velocity. AI experimentation moves quickly, and model selection often changes during product development. Security review can be perceived as a bottleneck on research momentum. The practitioner must separate experimentation from production promotion: exploration can remain flexible, but production promotion requires provenance, integrity verification, license review, eval evidence, and explicit approval. The gate applies to production, not to experimentation.

The structural challenge is ownership fragmentation. Research teams select models. ML platform teams host them. Product engineering integrates them. Legal reviews licenses. GRC needs evidence. Security owns intake review. If no one explicitly owns the model intake path from selection to production, artifacts move through informal trust channels. Supply chain security requires a defined handoff with explicit ownership at each stage.

The technical challenge is that model artifacts are often not self-describing. A checkpoint may not reveal its training data, base model lineage, or license implications. Some artifacts require custom loading code that must be separately reviewed. The practitioner must design an intake process that handles incomplete information explicitly — not by assuming that missing provenance means the artifact is safe, but by requiring provenance documentation as a prerequisite for production eligibility.

HOW TO APPROACH IT

- Define the production promotion trigger. Any model, adapter, embedding model, reranker, tokenizer, or preprocessing artifact that influences production behavior must enter a formal intake path. The trigger is production influence, not deployment to a production environment. An adapter that changes production model behavior must be intake-reviewed even if it is served through an existing production inference endpoint.

- Establish controlled artifact sources. Define which sources are approved for production artifacts: internal research with documented provenance, vendor-delivered artifacts with delivery metadata, and approved public hubs with mandatory intake review. For public hub downloads, the process should mirror the artifact into controlled internal storage after hash verification and approval — not pull from the hub directly at deployment time. Production deployments should not have live dependencies on mutable external sources.

- Design the intake record requirements. Each intake record should capture: owner, intended use, source URL, version identifier, artifact hash, base model name and version, fine-tuning process summary if applicable, license review outcome, allowed-format determination, eval evidence reference, security review status, approval record, deployment targets, and rollback version. These fields become the provenance record for the artifact's entire production lifetime.

- Build registry promotion as a technical control. Configure registry stages so that promotion to production-eligible stages requires: completed intake record with required fields, artifact hash match, license review completion, eval evidence reference, and explicit approver action. Access controls prevent arbitrary users from promoting to production stages. Registry promotion events generate audit records. The registry becomes the system of record for supply-chain evidence.

- Integrate checks into deployment pipelines. Deployments should: reject mutable artifact references and require pinned version identifiers, verify that the artifact hash matches the approved registry entry, confirm that required metadata is present, enforce format policy by blocking prohibited serialization formats, and record the exact artifact hash and registry entry loaded by each production service at each deployment.

OUTPUTS AND DELIVERABLES

- The intake artifacts are the **model intake record template**, **provenance record schema**, and **base model lineage map**. The intake record captures the required fields for production eligibility. The provenance schema defines the minimum documentation required for each artifact class: base models, fine-tunes, adapters, embedding models, and tokenizers. The lineage map makes inherited risk visible for fine-tuned and adapted models.

- The governance artifacts are the **model registry promotion policy**, **allowed format policy**, and **artifact integrity verification workflow**. The promotion policy defines required metadata, approval stages, access controls, rollback requirements, and evidence gates for each registry stage. The format policy categorizes each serialization format as permitted, permitted with sandboxing, or prohibited. The integrity workflow defines when hash verification runs, where approved artifacts are stored, and how production deployments prove they loaded the approved artifact.

- The release artifacts are the **model deployment manifest**, **supply chain CI/CD check specification**, and **license review record template**. The deployment manifest records the exact artifact hash, registry entry, eval evidence reference, owner, and rollback version for each production service. The CI/CD check specification defines automated checks that run during deployment. The license review record documents commercial rights, restrictions, and output implications for each production-eligible artifact.

COMMON FAILURE MODES

- **Hub-as-Trusted-Source:** The team deploys models directly from public hubs, treating hub publication as implicit provenance documentation. No hash verification, no intake review, no license review. Fix: require hub artifacts to mirror into controlled internal storage after intake review before any production reference. The production system never pulls directly from an external source.

- **Format-Safety-as-Supply-Chain:** The team migrates to safetensors and considers supply chain security complete. Provenance, license review, registry governance, and version pinning remain unaddressed. Fix: treat format safety as one control in a supply chain program, not as a substitute for the others.

- **Registry-as-Storage:** The model registry stores artifacts and makes them discoverable, but has no access control, no metadata requirements, no approval gates, and no audit records. Any team member can promote any artifact to production. Fix: configure the registry as a governance control with enforced metadata, defined promotion gates, access control, and audit logging.

- **Provenance Reconstruction Under Pressure:** When a security question arises about a production model, the team attempts to reconstruct provenance from model cards, git history, and team memory. The reconstruction is incomplete and unreliable. Fix: require provenance documentation before production promotion, not as incident response.

IMPLEMENTATION CHECKLIST

- [ ] Define the production promotion trigger for each model artifact class.
- [ ] Establish approved artifact sources and prohibit direct production pulls from external mutable references.
- [ ] Specify the intake record required fields for each artifact class.
- [ ] Configure the model registry with access control, metadata requirements, approval gates, and audit logging.
- [ ] Specify and enforce an allowed format policy with clear categorization for each serialization format.
- [ ] Build artifact hash verification into deployment pipelines with defined failure behavior.
- [ ] Require license review as a prerequisite for production promotion with a documented record.
- [ ] Generate deployment manifests that record the exact artifact hash and registry entry for each production deployment.

RELATED READING

- Handbook chapters: Chapter 8 (Model and Provider Risk) for externally hosted model provider risk management; Chapter 13 (Evaluation and Regression Testing) for eval evidence requirements at intake; Chapter 1 (AI System Inventory) for model dependency tracking.
- Field Guide: Model Supply Chain Security for provenance checks, integrity evidence, registry review, and license notes.