INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-22 13:52 UTC

Agentic AI Supply Chain Governance Gap: AI BOMs Emerge as Critical CISO Control

GOVERNANCE | MEDIUM | CVSS 5.0

SCC Item ID	SCC-GOV-2026-0037
Type	Governance
Severity	MEDIUM
CVSS Base Score	5.0
Affected Products	Organizations deploying agentic AI systems across enterprise environments, no specific product or version; broadly applicable
Published	2026-05-21T17:11:40
Discovery Source	Rss

Executive Summary

Organizations deploying agentic AI systems face a governance gap: existing software inventory frameworks were not designed to handle the dynamic, multi-component nature of autonomous AI agents, spanning third-party foundation models, fine-tuned layers, plugins, and runtime tool permissions, creating supply chain blind spots with no standardized documentation practice to close them. AI Bill of Materials (AI BOM) frameworks are emerging as the primary control response, designed to track model provenance, training data lineage, and runtime action scope in ways traditional SBOMs cannot. Without AI BOMs in place, security teams cannot determine whether a compromised upstream model or plugin has reached production, cannot scope incidents involving autonomous agent actions, and cannot demonstrate compliance with emerging AI governance obligations.

Technical Analysis

Agentic AI architectures introduce three distinct risk vectors absent from conventional software supply chain models. First, reliance on third-party foundation models and fine-tuned layers with no integrity verification creates exposure mapped to CWE-494 (Download of Code Without Integrity Check) and CWE-1104 (Use of Unmaintained Third-Party Components). Second, agent-granted tool permissions and API scopes frequently exceed least-privilege boundaries, mapped to CWE-284 (Improper Access Control). Third, dynamically loaded plugins and model artifacts lack runtime integrity checks, enabling potential hijack of agent execution paths. MITRE ATT&CK techniques relevant to this surface include T1195/T1195.001/T1195.002 (Supply Chain Compromise), T1574 (Hijack Execution Flow), T1078 (Valid Accounts, agent credential and permission abuse),

T1059 (Command and Scripting Interpreter, agent-executed code), and T1567 (Exfiltration Over Web Service via agent API calls). No CVE is associated; this is a systemic architectural and governance risk category. No patch exists; remediation requires process and documentation controls. Note: authoritative NIST and CISA guidance on AI BOM standardization is still emerging as of this writing, organizations should monitor NIST AI RMF updates for formalized standards.

Action Checklist

1. Inventory: Audit all production agentic AI deployments to identify third-party foundation models, fine-tuned layers, plugin integrations, and external API dependencies, apply CIS 1.1 (Enterprise Asset Inventory) and CIS 2.1 (Software Inventory) extended to AI components; document model versions, provenance, and update cadence.
2. Access Control Review: List all tool permissions, API scopes, and memory access grants assigned to each agent, verify alignment with least privilege per NIST AC-6 (Least Privilege) and CIS 5.4 (Restrict Administrator Privileges); revoke permissions not required for defined agent task scope.
3. Integrity Verification: Implement cryptographic integrity checks on all dynamically loaded model artifacts and plugins before execution, address CWE-494 directly; where vendor-supplied checksums or signing are unavailable, document as an unmitigated risk and escalate to vendor.
4. Detection Baseline: Enable audit logging on all agent tool invocations, API calls, and permission escalation events per NIST AU-2 (Event Logging) and AU-12 (Audit Record Generation); route logs to SIEM with alerting on anomalous tool use, unexpected external API destinations, and permission scope changes.
5. Governance Documentation: Draft or adopt an AI BOM template for each production agentic system documenting: model weights source and version, training data lineage (where known), third-party API dependencies, plugin manifest, runtime tool permissions, and update/change history, align to NIST AC-20 (Use of External Systems) for third-party model dependencies and NIST CM controls for configuration management of AI components; schedule quarterly review cycles per CIS 7.1 (Vulnerability Management Process).

IR / Forensic Enrichment

Triage Priority	STANDARD
Escalation Criteria	Escalate to urgent if the inventory step (Step 1) discovers agentic AI systems processing PII, PHI, or financial data that lack any provenance documentation or integrity controls, triggering potential breach notification obligations under GDPR Article 33, HIPAA §164.410, or state privacy laws due to inability to bound data exposure scope.
Recovery Notes	After access controls are tightened and integrity checks are operational, re-run each production agent against its defined task scope in a staging environment to verify that least-privilege permission revocations have not broken legitimate functionality before returning to production. Monitor agent tool invocation logs for 30 days post-remediation for behavioral drift — specifically, watch for agents attempting to call previously-revoked API scopes, which would indicate either misconfiguration or that the agent's underlying prompt or model is attempting to re-acquire permissions. Schedule the first quarterly AI BOM review at 90 days and treat any newly discovered undocumented components as a governance incident requiring root cause analysis.

Forensic Artifacts	Agent framework configuration files (LangChain agent YAML/JSON, AutoGPT config.yaml, CrewAI crew definitions, Semantic Kernel skill manifests) — these contain the authoritative record of what tools and permissions were defined at deployment time and are the primary artifact for reconstructing pre-incident agent capability scope. | Model weight file checksums and download provenance logs (pip install history at '~/.local/share/pip/log/', Hugging Face cache at '~/.cache/huggingface/hub/' including 'model.safetensors.metadata' files) — tampering or unexpected model substitution would appear as a hash mismatch between cached artifacts and the original download record. | External API call logs from the agentic AI runtime (OpenAI API request logs via '/v1/usage', Anthropic API logs, plugin API gateway access logs) — unexpected destination domains, unusual call volumes, or API calls to endpoints not listed in the plugin manifest are the primary behavioral indicators of prompt injection or tool abuse. | Plugin manifest.json files and their declared permission scopes versus the OAuth token introspection responses from each connected service — a delta between declared permissions and granted permissions indicates either misconfiguration or unauthorized scope escalation that occurred outside the formal provisioning process. | Vector store access logs and memory namespace audit trails (Chroma collection access logs, Pinecone index query history, Redis vector DB command logs) — cross-agent memory pollution or unauthorized reads of other agents' memory namespaces would appear as queries from unexpected agent_id values against collections outside the querying agent's documented scope.

Per-Action IR Details

Inventory: Audit all production agentic AI deployments to identify third-party foundation models, fine-tuned layers, plugin integrations, and external API dependencies — apply CIS 1.1 (Enterprise Asset Inventory) and CIS 2.1 (Software Inventory) extended to AI components; document model versions, provenance, and update cadence.

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Establishing IR capability requires knowing what assets exist; agentic AI components (foundation models, fine-tuned adapters, plugin manifests) are undocumented assets that create blind spots before any incident occurs.

Controls: CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), NIST CM-8 (System Component Inventory), NIST SA-9 (External System Services)

Compensating: Run 'pip list --format=json' or 'conda list --json' on all AI workload hosts to capture installed model libraries and versions; supplement with 'find / -name "*.gguf" -o -name "*.safetensors" -o -name "*.bin" 2>/dev/null' to locate locally cached model weights. For containerized agents, execute 'docker inspect ' and parse the Env and Labels fields for model endpoint URIs and API keys. Maintain results in a versioned CSV or Git-tracked YAML file as your AI BOM baseline.

Evidence: Before inventorying, preserve the current state: snapshot container image digests ('docker images --digests'), capture running process trees ('ps auxf' or 'Get-Process | Select-Object Id,Name,Path | Export-Csv'), and export any existing orchestration config files (LangChain YAML, AutoGPT config.yaml, CrewAI agent definitions) to a read-only evidence directory. These establish a pre-audit baseline to detect changes introduced during the inventory process itself.

Access Control Review: Enumerate all tool permissions, API scopes, and memory access grants assigned to each agent — verify alignment with least privilege per NIST AC-6 (Least Privilege) and CIS 5.4 (Restrict Administrator Privileges); revoke permissions not required for defined agent task scope.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: Reducing the blast radius of a compromised or misbehaving agentic AI system by constraining its tool permissions and API scopes mirrors network-level containment; over-permissioned agents can exfiltrate data, invoke destructive tools, or pivot to internal systems before detection.

Controls: NIST AC-6 (Least Privilege), NIST AC-3 (Access Enforcement), NIST AC-4 (Information Flow Enforcement), NIST AC-17 (Remote Access), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts), CIS 6.1 (Establish an Access Granting Process), CIS 6.2 (Establish an Access Revoking Process)

Compensating: Export all OAuth 2.0 token scopes and API key permissions from agent configuration files (e.g., LangChain tool definitions, OpenAI function-calling schemas, plugin manifest.json files) and paste into a spreadsheet mapping each permission to a specific named agent task. For AWS Bedrock or similar managed AI services, run 'aws iam simulate-principal-policy' against the agent's IAM role to identify permissions granted but never invoked. Immediately rotate any API keys found with write, delete, or admin scopes that are not explicitly required by the documented agent task definition.

Evidence: Before revoking permissions, capture the full current permission state as forensic baseline: export all agent IAM role policies ('aws iam get-role-policy'), OAuth token introspection responses, and plugin manifest.json files with their declared 'permissions' arrays. For memory-enabled agents (e.g., those using vector stores like Chroma or Pinecone), document which collections each agent has read/write access to and whether cross-agent memory namespacing is enforced. This evidence establishes the pre-containment attack surface if a future incident requires reconstruction of what an agent could have accessed.

Integrity Verification: Implement cryptographic integrity checks on all dynamically loaded model artifacts and plugins before execution — address CWE-494 directly; where vendor-supplied checksums or signing are unavailable, document as an unmitigated risk and escalate to vendor.

NIST Phase: Preparation

Reference: NIST 800-61r3 §2 — Preparation: Integrity verification of AI model artifacts is a pre-incident control; CWE-494 (Download of Code Without Integrity Check) in the agentic AI context means a tampered model weight file or malicious plugin could execute arbitrary behavior before any detection capability fires.

Controls: NIST SI-7 (Software, Firmware, and Information Integrity), NIST SA-12 (Supply Chain Protection), NIST CM-3 (Configuration Change Control), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: Generate SHA-256 checksums for all model weight files at time of initial download ('sha256sum model.safetensors > model.safetensors.sha256') and store hashes in a Git repository separate from the model storage location. Implement a pre-launch shell script that recomputes the hash and compares against the stored value, exiting non-zero and alerting if mismatch is detected. For Hugging Face models, validate against the published 'sha256' field in the model card's 'model-index' metadata using 'huggingface-cli' or direct API call to 'https://huggingface.co/api/models/' — note that not all models publish checksums, which must itself be documented as an unmitigated CWE-494 instance.

Evidence: Before implementing integrity checks, collect current checksums of all existing model artifacts as a forensic baseline ('find /models -type f \(-name "*.safetensors" -o -name "*.bin" -o -name "*.gguf" \) -exec sha256sum {} \;') and compare against any available vendor-published hashes. Capture plugin manifest.json files and their declared 'api' endpoint URLs — a tampered plugin will show a hash mismatch or point to an unexpected domain. Preserve these baseline hashes in immutable storage (write-once S3 bucket or signed Git commit) before the verification workflow goes live.

Detection Baseline: Enable audit logging on all agent tool invocations, API calls, and permission escalation events per NIST AU-2 (Event Logging) and AU-12 (Audit Record Generation); route logs to SIEM with alerting on anomalous tool use, unexpected external API destinations, and permission scope changes.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Agentic AI systems have no native equivalent to Windows Event IDs or syslog standards; establishing what 'normal' tool invocation patterns look like for each agent is the foundational detection baseline, without which prompt injection, tool abuse, and unauthorized API pivoting are invisible.

Controls: NIST AU-2 (Event Logging), NIST AU-3 (Content of Audit Records), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: If no SIEM is available, deploy a local ELK stack (Elasticsearch + Logstash + Kibana, all free/open-source) and configure LangChain's callback handler ('BaseCallbackHandler') or equivalent framework hooks to emit structured JSON logs of every tool call, including: agent_id, tool_name, input_payload, output_summary, timestamp, and external_endpoint. Write a daily cron job using 'jq' to parse logs and flag: (1) tool invocations calling domains not in an approved allowlist, (2) permission escalation events where an agent requests a scope not in its original manifest, and (3) sequences where an agent invokes more than N tools in a single chain (configurable threshold). For Python-based agents, instrument with OpenTelemetry and export traces to a self-hosted Jaeger instance for call graph anomaly review.

Evidence: Before establishing the detection baseline, capture a representative sample of current agent execution traces during normal operations to define the behavioral baseline: export LangChain callback logs, OpenAI API usage logs (available via the API's '/v1/usage' endpoint), and any existing application logs showing tool call sequences. Document the approved external API destination list (domains, IPs, ports) that agents legitimately contact. This baseline is the comparison anchor for all future anomaly detection — without it, alert thresholds cannot be calibrated and every deviation will produce unactionable noise.

Governance Documentation: Draft or adopt an AI BOM template for each production agentic system capturing: model weights source and version, training data lineage (where known), third-party API dependencies, plugin manifest, runtime tool permissions, and update/change history — align to NIST AC-20 (Use of External Systems) for third-party model dependencies and NIST CM controls for configuration management of AI components; schedule quarterly review cycles per CIS 7.1 (Vulnerability Management Process).

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: AI BOM documentation functions as the lessons-learned artifact that closes the governance gap; each quarter's review cycle is equivalent to a post-incident review that asks whether the AI supply chain posture has degraded since the last assessment.

Controls: NIST AC-20 (Use of External Systems), NIST CM-3 (Configuration Change Control), NIST CM-8 (System Component Inventory), NIST SA-9 (External System Services), NIST RA-3 (Risk Assessment), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 3.2 (Establish and Maintain a Data Inventory)

Compensating: Maintain the AI BOM as a Git-tracked YAML or Markdown file per agent, with required fields: 'model_id', 'source_registry' (e.g., Hugging Face model ID or OpenAI model string), 'version_pinned' (exact version hash, not 'latest'), 'plugins' (array of plugin name + manifest URL + last_verified date), 'tool_permissions' (array matching the access control review output), 'external_apis' (domain + auth_method + data_sent), and 'last_reviewed'. Enforce quarterly review via a GitHub Actions workflow that opens a review issue automatically every 90 days and blocks merges to main if the 'last_reviewed' field is older than 91 days. Use OWASP's ML-BOM draft specification as the field schema reference until a formal standard is finalized.

Evidence: Before finalizing the AI BOM template, collect all existing documentation artifacts that should have captured this information but did not: architecture diagrams, vendor contracts referencing model providers, API key registration records, and any prior change management tickets for AI component updates. Gaps between what these documents describe and what the inventory step discovered (Step 1) constitute the documented governance gap that justifies the AI BOM program — preserve this delta as the risk register entry that drives quarterly review prioritization.

Detection Guidance

No IOC-based detection applies; this is a governance gap, not an active exploit campaign. Detection focus is on anomalous agent behavior and supply chain integrity signals. Log and alert on: (1) agent processes initiating outbound API calls to destinations not in an approved allowlist, cross-reference against NIST AC-4 (Information

Flow Enforcement) baselines; (2) permission or scope escalation events where an agent requests tool access beyond its defined task profile, monitor via NIST AU-6 (Audit Record Review) workflows; (3) dynamically loaded model artifacts or plugins that do not match a known-good hash registry, implement D3-FMBV (File Magic Byte Verification) and D3-SFA (System File Analysis) on AI artifact load events; (4) lateral tool invocations where an agent calls systems outside its designated workflow, flag via behavioral baselining in SIEM. For organizations using LLM orchestration frameworks (LangChain, AutoGen, CrewAI, or similar), enable verbose execution logging and pipe to a centralized log store. Query for: agent task chains exceeding defined step counts, tool calls to new or uncatalogued external services, and memory read/write events outside scoped data stores. Absence of these logs is itself a detection gap requiring remediation under NIST AU-2.

Framework Mappings

MITRE-ATTACK

- **T1078** — Valid Accounts
- **T1574** — Hijack Execution Flow
- **T1195.001** — Compromise Software Dependencies and Development Tools
- **T1195.002** — Compromise Software Supply Chain
- **T1059** — Command and Scripting Interpreter
- **T1195** — Supply Chain Compromise
- **T1567** — Exfiltration Over Web Service

NIST-800-53R5

- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CM-7** — Least Functionality
- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **SI-7** — Software, Firmware, and Information Integrity
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SR-2** — Supply Chain Risk Management Plan
- **CM-3** — Configuration Change Control
- **AC-3** — Access Enforcement
- **SA-4** — Acquisition Process

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures
- **A01:2021** — Broken Access Control
- **A06:2021** — Vulnerable and Outdated Components

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process
- **16.4** — Establish and Manage an Inventory of Third-Party Software Components
- **15.1** — Establish and Maintain an Inventory of Service Providers

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC9.2** — Manages risks associated with vendors and business partners

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.21** — Managing information security in the ICT supply chain

NIST-CSF-2

- **GV.SC-01** — Cybersecurity supply chain risk management program

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1078	Valid Accounts	Defense-Evasion
T1574	Hijack Execution Flow	Persistence
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access
T1195.002	Compromise Software Supply Chain	Initial-Access
T1059	Command and Scripting Interpreter	Execution
T1195	Supply Chain Compromise	Initial-Access
T1567	Exfiltration Over Web Service	Exfiltration

Sources

Source	URL	Tier
Security News	https://www.darkreading.com/cyber-risk/how-cisos-should-prep-for-ag...	T3
American and Allied Cyber Agencies Issue First Joint Guidance on ...	https://www.crowell.com/en/insights/client-alerts/american-and-alli...	T3
Agentic AI: How It Works and 7 Real-World Use Cases | Exabeam	https://www.exabeam.com/explainers/ai-cyber-security/agentic-ai-how...	T3
Understanding Agentic AI Risks - Cloud Security Alliance (CSA)	https://cloudsecurityalliance.org/blog/2025/05/12/agentic-ai-unders...	T3
Transforming cybersecurity with agentic AI to combat emerging ...	https://www.sciencedirect.com/science/article/pii/S0308596125000734	T3

Generated 2026-05-22 13:52 UTC by TJS Security Command Center