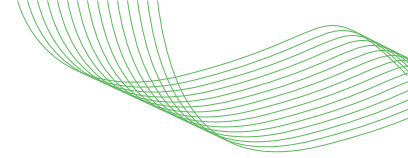




# Establish Risks and Controls for the AI Supply Chain, V 1.0

---

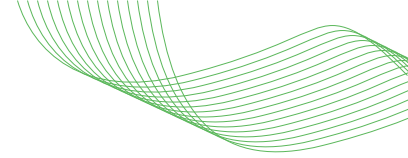
Workstream 1: Software Supply Chain Security for AI Systems



# Contents

---

<b>Establish Risks and Controls for the AI Supply Chain, V 1.0</b>	<b>2</b>
Executive Summary . . . . .	2
1. Introduction . . . . .	2
1.1. What's different about AI? . . . . .	2
2. Defining the AI Supply Chain . . . . .	3
2.1 Model Generation . . . . .	4
Supply Chain Documentation Requirements . . . . .	10
2.3 Scope of Existing Secure AI Frameworks . . . . .	11
3. Risks, Threats, and Existing Mitigations . . . . .	11
3.1 Model Generation . . . . .	11
3.1.1 Data Poisoning: Threats and Mitigations in AI Supply Chains . . . . .	11
3.1.2 Model Training Security: Risks and Mitigations . . . . .	13
3.1.3 Application Integration in the AI Supply Chain . . . . .	15
3.1.4 AI Infrastructure Supply Chain Security . . . . .	16
3.2 Model Integration and Consumption . . . . .	19
3.2.1 Supply Chain Security for Data . . . . .	19
3.2.2 Model . . . . .	22
3.2.3 Application . . . . .	24
3.2.4 Infrastructure . . . . .	26
4. Key Takeaways for Stakeholders in Your Organization . . . . .	29
4.1 Executive Leadership . . . . .	29
4.2 Security Practitioner . . . . .	30
4.3 AI Researcher/Engineer . . . . .	31
5. Conclusion . . . . .	32
6. Contributors and Acknowledgements . . . . .	33
7. Appendix . . . . .	34
7.1 CoSAI Focus . . . . .	34
7.2 Guidelines on usage of more advanced AI systems (e.g. large language models (LLMs), multi-modal language models. etc) for drafting documents for OASIS CoSAI: . . . . .	34
7.3 Copyright Notice . . . . .	35



# Establish Risks and Controls for the AI Supply Chain, V1.0

---

OASIS Open Project : Coalition for Secure AI (CoSAI) Workstream 1: Software Supply Chain Security for AI Systems

*Approved by the CoSAI Project Governing Board on 12 June 2025*

## Executive Summary

The rapidly evolving artificial intelligence landscape presents unique supply chain security challenges that traditional security measures alone cannot address. AI systems require specialized approaches to mitigate sophisticated emerging risks across their entire supply chain, encompassing data, infrastructure, models, and applications.

Key security considerations include comprehensive risk assessment spanning the entire AI pipeline from data acquisition to model deployment, provenance tracking to ensure integrity and transparency of data sources and model origins, and specialized security protocols for AI-specific vulnerabilities during model generation and integration. Equally important are data integrity protection throughout the collection, processing, and training phases, secure training environments to prevent contamination or manipulation of models, and robust vulnerability management in AI-powered applications and integrated systems.

The paper examines the critical roles of executive leadership, security practitioners, and AI researchers in implementing effective risk mitigation strategies. It also compares leading AI security frameworks, evaluating their strengths and limitations specifically for supply chain security.

Effective AI supply chain security requires continuous monitoring, robust protocols, and collaborative efforts among all stakeholders to ensure the safe, ethical deployment of AI technologies within complex software ecosystems.

NOTE: this document is intended as a living resource. The Coalition for Secure AI (CoSAI) commits to continually updating its content to reflect emerging technologies, new threat vectors, and evolving best practices, ensuring that our guidance remains current and actionable for all stakeholders.

## 1. Introduction

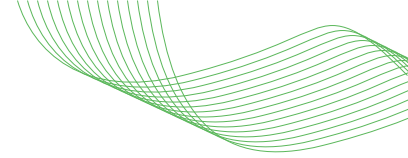
Threat modeling is a critical process for ensuring the security and resilience of AI systems. Of particular importance are the threats and security of the software supply chain for these AI systems. While threat modeling is a mature discipline and common practice in developing software, AI systems present distinct challenges and considerations.

### 1.1. What's different about AI?

To understand the AI software supply chain better, we must first examine how AI applications differ fundamentally from traditional software applications.

At the core of AI systems are artificial neural networks containing hundreds of millions of nodes, each with associated weights that determine how the network processes input data. These weights—generated during the training phase from large datasets—embody the “intelligence” of the system. Unlike traditional applications where sensitive information resides in specific memory locations, in AI systems, knowledge is distributed across these countless weighted connections, creating a unique security paradigm.

This distributed architecture creates multiple attack surfaces throughout the AI supply chain:



**Data Supply Chain Vulnerabilities:** Attackers can poison training data through direct manipulation or by injecting malicious content onto internet pages that will later be crawled. This poisoning need not be sophisticated—carefully crafted examples can create subtle biases or backdoors that activate only under specific conditions, altering the weights in ways that benefit the attacker while remaining difficult to detect.

**Model Supply Chain Vulnerabilities:** Pre-trained models transferred between organizations present significant risks. Attackers can take existing models and fine-tune them with poisoned data, subtly altering their behavior. The weights themselves may be directly tampered with during transfer or storage—changes to a few weights might cause little noticeable effect, but at some critical threshold, they can dramatically impact model behavior. These manipulations often evade detection while fundamentally altering how the model responds to inputs.

Reference documents or resources used to provide context during inference can contain hidden adversarial prompts. Instructions to extract and exfiltrate sensitive information might be buried in reference materials or hidden stenographically in images. Unlike traditional security threats requiring network penetration, these attacks exploit the system’s intended functionality by manipulating the contextual information it processes.

**Application Supply Chain Vulnerabilities:** The integration of AI components into larger systems creates opportunities for exploitation. Models developed by individuals with limited security awareness might be integrated into critical applications like banking, healthcare, or employment systems without proper security validation. These compromised models could leak sensitive data or generate falsified outputs, such as creating fake credit scores, exposing personal health information, or manipulating employment background checks.

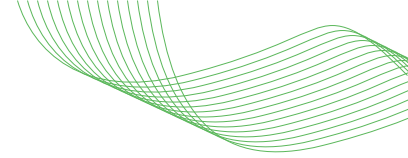
What makes these supply chain attacks particularly concerning is that attackers no longer need to target specific code or memory locations—they can influence AI systems by using their learning and processing mechanisms against them. Moreover, these attacks don’t necessarily require technical hacking expertise; they simply require understanding how to craft inputs that manipulate the model’s behavior.

The complex web of actors involved in this multi-layered supply chain complicates accountability. With numerous stakeholders involved in data collection, model development, application integration and infrastructure provisioning, determining responsibility when failures occur becomes increasingly difficult. This diffusion of responsibility creates significant challenges for establishing clear security standards, attributing blame, and developing appropriate legal liability frameworks for AI security breaches.

## 2. Defining the AI Supply Chain

The AI supply chain is a multifaceted ecosystem encompassing four critical dimensions: Data, Model, Application, and Infrastructure. Each of these components plays a vital role in the development, deployment, and security of AI systems. Data serves as the foundation, influencing model performance and trustworthiness, while the model itself—comprising its architecture, training data, and fine-tuning processes—represents the core intelligence behind AI-driven solutions. The application layer integrates AI models into end-user products, ensuring functionality and interaction with external systems while the infrastructure encompasses the computing resources, storage, and networking that support AI workloads. A comprehensive understanding of these four dimensions is essential to securing the AI lifecycle, as vulnerabilities in any stage can propagate throughout the entire system.

The following sections focus on two key stages: (a) Model Generation and (b) Model Integration and Consumption. Model Generation explores how AI models are trained, fine-tuned, and stored, while Model Integration and Consumption examines how these models interact with applications and users. Both stages present unique security challenges that must be addressed to ensure a



resilient AI supply chain.

## 2.1 Model Generation

Model generation encompasses traditional software components (serving stacks, application code) alongside specialized data requirements for training. Each component introduces potential risk vectors that security practitioners and data scientists must meticulously monitor and document to maintain supply chain integrity throughout the AI development lifecycle.

### The Generative AI Landscape

Today's AI landscape is dominated by Generative AI, particularly Large Language Models (LLMs). These models typically progress through at least three distinct stages:

1. Initial training of foundation models on extensive text corpora (often petabytes of data)
2. Fine-tuning steps for specific tasks through techniques like RLHF (Reinforcement Learning from Human Feedback)
3. Deployment into applications via inference APIs or embedded runtimes

Different teams typically handle each stage, with models stored in separate infrastructure between stages. This storage infrastructure itself represents a critical part of the AI supply chain, often involving specialized formats (safetensors, GGUF, ONNX) and large binary weights files. For example, misconfigured storage permissions or inadequate cryptographic signing could allow a malicious insider to manipulate model weights, compromising numerous downstream applications without detection.

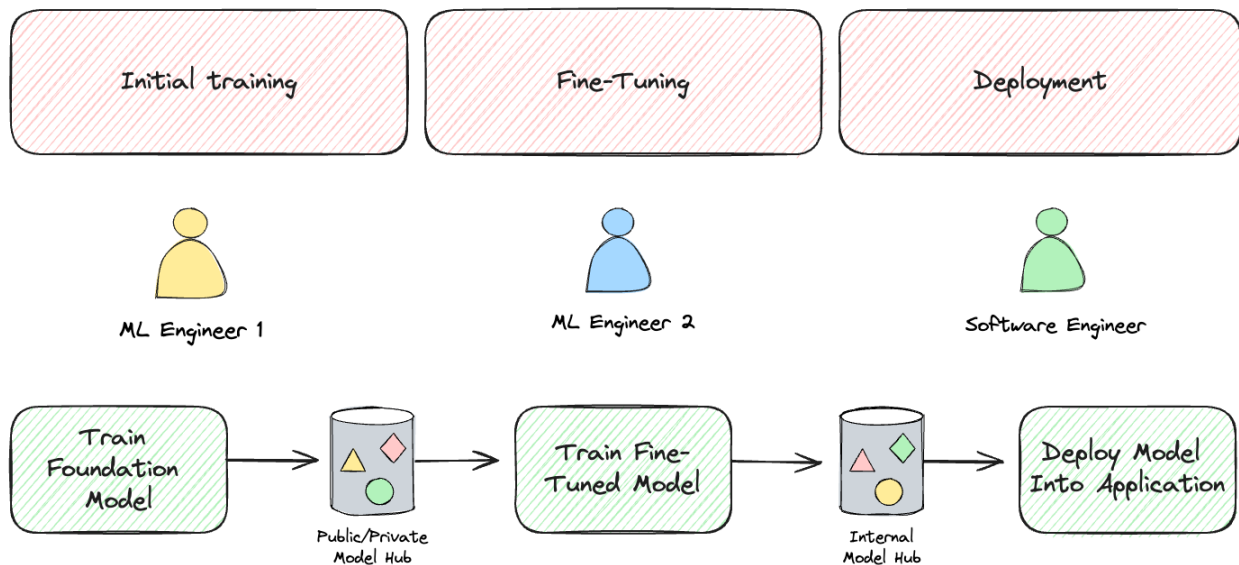


Figure 1: *lifecycle*

### The Model Development Process

Model development begins with data collection from diverse sources across the internet. This data requires thorough cleaning, filtering, and processing to eliminate duplicates, correct misinformation, and address missing values. This iterative process may employ both manual effort and specialized preprocessing pipelines implementing techniques like tokenization, normalization, and deduplication algorithms.

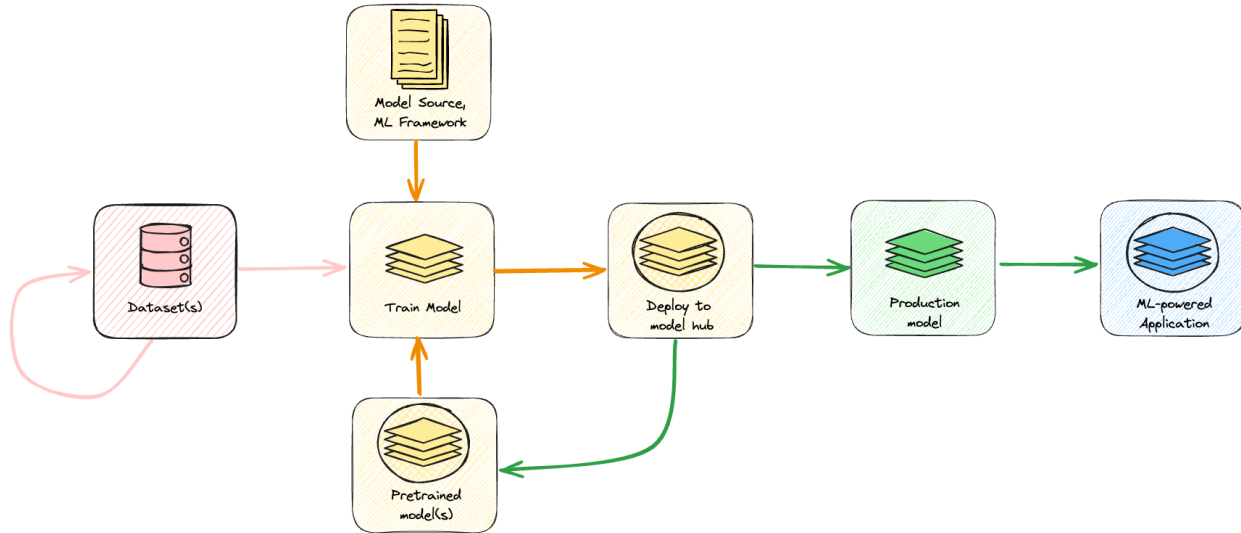


Figure 2: *deployment*

Data quality is paramount as models inherit the properties of their training data, including biases, factual errors, or security vulnerabilities. Consequently, data represents a prime target for supply chain attacks, where adversaries might poison datasets to introduce instability or manipulate model behavior. Advanced attackers may employ techniques like backdoor injection or gradient-based poisoning that remain undetectable in standard quality assessments.

### Broader Supply Chain Considerations

A holistic view of AI supply chains must include applications used in model training. Vulnerabilities may exist in:

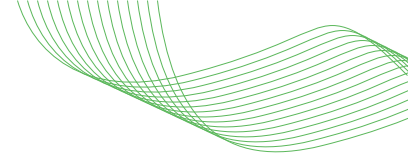
- Data gathering applications, such as web crawlers and scrapers
- Sorting tools and feature extraction pipelines
- Storage infrastructure, including distributed file systems (e.g., HDFS, S3)
- Containerized training environments and orchestration systems (e.g., Kubernetes)

For example, vulnerabilities in data cleaning applications could compromise datasets, poisoning the ML supply chain at its source. A compromised package in a Python dependency could silently modify training data. Anyone with write access to data storage could manipulate training data through subtle alterations that optimize for adversarial outcomes. Therefore, access to critical data and resulting models must be strictly limited, monitored, and controlled through comprehensive identity management, audit logging, and integrity verification mechanisms.

### Configuration and Training Parameters

LLM training typically involves dataset configuration via mixture weights (e.g., 80% from high-confidence sources, 15% from trusted news, 5% from forums). These weights are implemented through sampling strategies in data loaders and batch formation algorithms. Since these mixtures significantly influence model performance, their parameters must be recorded cryptographically to prevent manipulation that could bias outputs toward specific ideological positions or harmful behaviors.

A comprehensive governance structure should include a manifest detailing:



- Data sources with precise version identifiers
- Points of origin (URLs, repositories, data brokers)
- Data attributes including checksums and content hashes
- Contents and usage statistics including token counts and distribution metrics
- Preprocessing transformations applied to raw data

This training data provenance should extend to validation and fine-tuning datasets, enabling potential third-party audits and increasing consumer confidence. Solutions like data lineage tools, content-addressed storage, and cryptographic attestations can verify the integrity of this information throughout the model lifecycle.

### Model Architecture and Frameworks

Model development requires defining the architecture (layers, transformer blocks, attention mechanisms, etc.) using APIs specified by the chosen ML framework (PyTorch, TensorFlow, JAX). Both the model source code and the ML framework itself introduce supply chain considerations:

- Framework vulnerabilities in optimization routines could prevent proper weight updates or introduce numerical instabilities
- Source code alterations could insert architectural backdoors triggered by specific inputs, such as specially crafted prompts
- Hyperparameter manipulation could create models that generalize poorly or exhibit specific weaknesses
- Pre-compiled binaries for accelerated hardware may contain compromised code

All code dependencies should be pinned to specific versions with verified integrity hashes and scanned for vulnerabilities before deployment in training infrastructure.

### Pre-trained Components and Checkpoints

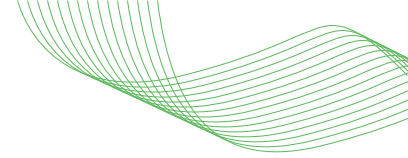
Training may leverage previous checkpoints or incorporate separate pre-trained models through techniques like parameter-efficient fine-tuning (LoRA, QLoRA) or knowledge distillation. Their provenance must be documented for:

1. Completeness in supply chain analysis, including transitive dependencies
2. Governance and cost control, as training requires significant computational resources (often measured in GPU/TPU-hours)
3. Security assurance through verifiable builds and signed model artifacts

All input models/checkpoints and those generated during training require secure storage with integrity verification. Checkpoints should be cryptographically signed and verified before use in subsequent training stages. As noted previously, storage represents a common vector for insider threats, requiring comprehensive access controls and tamper-evident logging.

### Evaluation Processes

Post-training evaluation forms a critical part of the ML supply chain. Inadequate evaluation processes can result in vulnerabilities when models are deployed. Evaluation infrastructure and code defining the evaluation processes require:



- Diverse benchmark datasets to assess model performance across different dimensions
- Adversarial testing frameworks to identify potential security vulnerabilities
- Ethical red-teaming to probe for harmful outputs or behaviors
- Automated testing for common failure modes and prompt injection attacks

Each evaluation component represents additional supply chain elements requiring security consideration, with their own dependencies and potential attack surfaces.

### **API Profiling for Enhanced Security**

API profiling provides an additional security layer through runtime monitoring techniques. By tracking framework API calls during model development stages (training, fine-tuning, validation), organizations can establish expected behavior profiles using statistical baselines and permitted call sequences. These profiles can later detect anomalous API activity potentially indicating compromise.

In a practical example, consider a properly trained model with comprehensive API profiles across all development stages. If an adversary attempts to manipulate the model into divulging private data through prompt injection, unexpected API calls to file system or network interfaces might occur. Comparison against the model's validation API profile would flag this anomalous behavior through statistical deviation detection, making it easier to detect attacks and harder for adversaries to subvert the model.

API profiling can be implemented through instrumentation of the inference runtime, sandbox environments with controlled capabilities, and continuous monitoring systems that compare actual behavior against established baselines, providing an additional defense-in-depth measure against sophisticated attacks.

## **2.2 Model Integration and Consumption The End-User Perspective**

From an end-user's perspective, AI-powered applications represent a unified system combining the AI model with application infrastructure. While the interaction appears straightforward—user submits a prompt, the application returns a response—the underlying architecture contains numerous components, each introducing potential security vulnerabilities into the supply chain.

### **Input-Output Processing Pipeline**

Behind the user interface, sophisticated processing occurs at multiple stages that introduce various security considerations.

#### **Input/Output Filtering Systems**

Applications implement filtering mechanisms for both inputs and outputs to create a secure interaction environment. These systems enforce safety guardrails through prompt classification and content detection algorithms while applying privacy protection through PII detection and redaction techniques. Modern filtering systems also prevent prompt injection attacks via pattern matching and heuristic analysis, and implement content moderation using specialized filtering models designed to detect harmful content patterns.

Each filtering component represents a discrete element in the supply chain with its own dependencies, update cadence, and potential for compromise. An attacker who gains control of these filtering mechanisms could selectively bypass security controls for targeted users or queries, potentially enabling data exfiltration or harmful content generation while appearing benign to security monitoring systems. This makes the integrity of filtering components particularly critical to the overall security posture of AI applications.

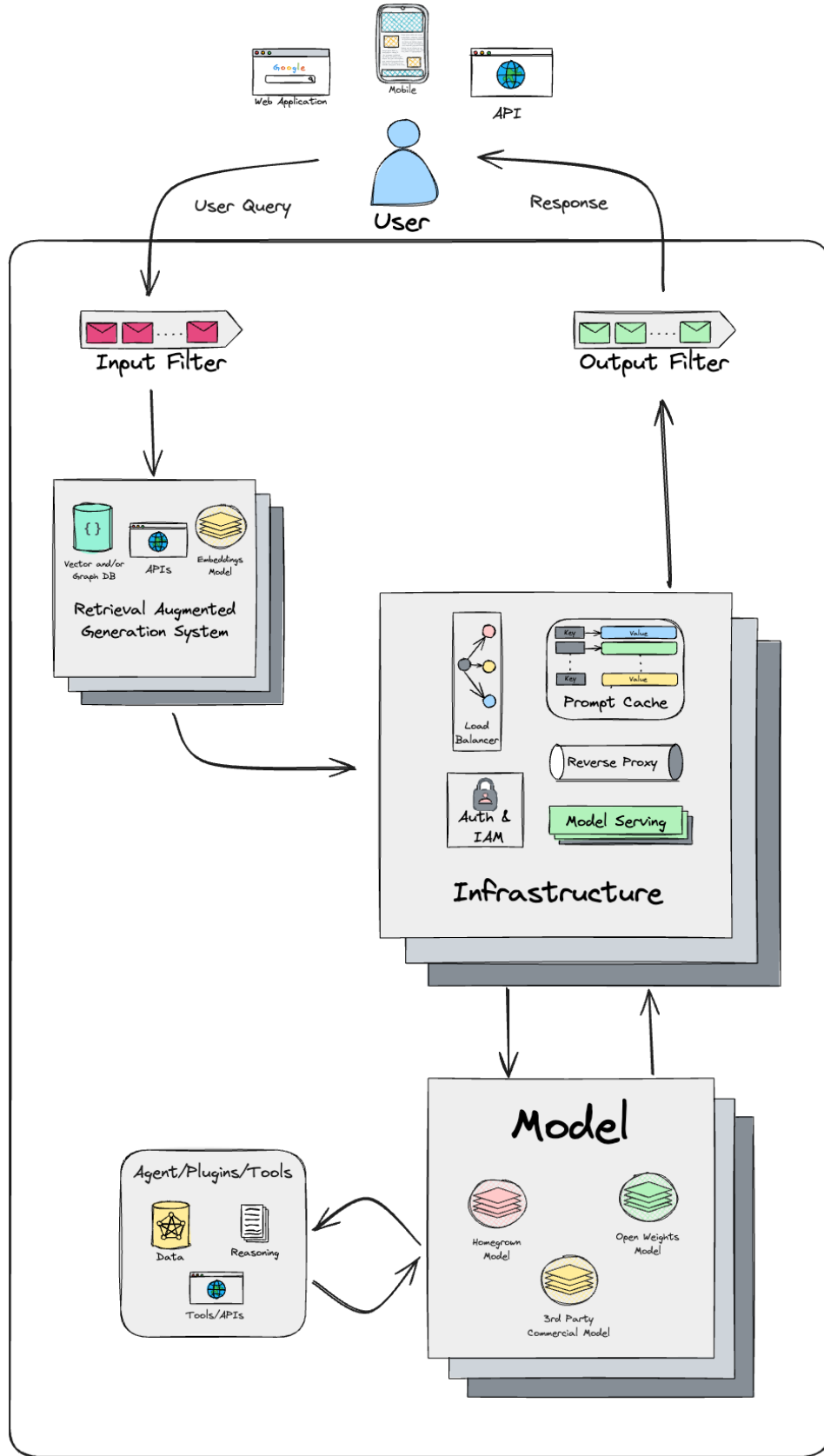
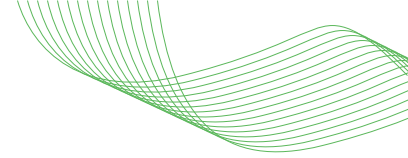


Figure 3: architecture



## Context Enrichment and RAG Systems

After passing through input filters, queries undergo enrichment with application-specific context through increasingly common Retrieval Augmented Generation (RAG) pipelines. These systems match user queries against enterprise data stored in vector databases such as Pinecone, Milvus, or Weaviate. The semantic search algorithms determine relevance through cosine similarity or other distance metrics, while document chunking and embedding strategies influence which information gets retrieved. Context window management systems then determine what information fits within model token limits, making crucial decisions about what context reaches the model.

The RAG infrastructure consists of several interdependent components:

1. Embedding models that transform text to vector representations
2. Vector databases that store and index these embeddings
3. Retrieval algorithms that perform approximate nearest neighbor searches
4. Context assembly systems that format retrieved information for the model

Each component represents a potential attack surface where an adversary could inject malicious data or modify retrieval behavior. For instance, poisoning embedding models could cause retrieval of incorrect or harmful information for specific queries, while compromised vector databases might leak sensitive information across tenant boundaries in multi-tenant environments. Organizations must consider these components as part of their AI supply chain security strategy and implement appropriate controls to ensure their integrity.

## Model Serving Infrastructure

The model serving layer introduces additional supply chain components that require security consideration, spanning authentication, scaling, and performance optimization systems.

### Authentication and Authorization Systems

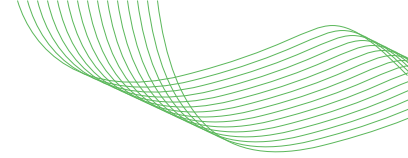
Identity and Access Management (IAM) services controlling model access form the first line of defense in the model serving infrastructure. These systems implement API key management with appropriate rotation mechanisms and token-based authentication frameworks to verify the identity of calling applications and users. Well-designed systems employ role-based access control (RBAC) implementations with fine-grained permissions models for different API operations, limiting access based on the principle of least privilege.

The security of these components is paramount, as compromised authentication systems could allow unauthorized access to models, potentially exposing sensitive data or enabling resource abuse. Organizations must ensure these systems undergo rigorous security testing and are kept updated with the latest security patches.

### Load Balancing and Scaling Infrastructure

Behind production AI systems lies complex infrastructure managing request distribution across model replicas. This infrastructure includes auto-scaling mechanisms based on traffic patterns and queue management systems for handling traffic spikes. Health checking and failover mechanisms ensure reliability while maintaining security boundaries.

These components must be secured against denial-of-service attacks and configured to prevent unauthorized access to model instances. Improper configuration could lead to resource exhaustion or enable attackers to bypass security controls by targeting specific model replicas with known vulnerabilities.



### Performance Optimization Systems

Prompt caching systems accelerate response times by storing results of previous queries, but introduce significant security concerns that must be carefully managed. Cache poisoning vulnerabilities could affect multiple users if an attacker can manipulate cache entries. Side-channel attacks might extract information about other users' queries by analyzing cache hit patterns, while cache timing attacks could reveal sensitive patterns in system behavior.

Perhaps most concerning is the risk of improper multi-tenant isolation, which could leak data across organizational boundaries if cache entries from one tenant become visible to another. These caching mechanisms must implement proper isolation, encryption, and cache entry validation to prevent cross-user data leakage and ensure cache integrity throughout the system life-cycle.

### Inference-Time Components

During inference, models may access additional systems to enhance responses, creating a complex web of dependencies with security implications.

### Augmentation Components

AI systems increasingly rely on external components to enhance their capabilities. External knowledge bases and proprietary databases provide factual information, while reasoning engines implement logical operations that extend model behavior. Function calling frameworks enable tool use for tasks like calculations or data retrieval, and plugin ecosystems further extend model capabilities into specialized domains. Some advanced systems employ multi-agent architectures orchestrating specialized AI components working in concert to solve complex problems.

Each component introduces its own supply chain dependencies and attack surfaces that must be considered. Tools with network access could be exploited for data exfiltration if improperly secured. Plugins with file system access could introduce remote code execution vulnerabilities if they fail to properly validate inputs. Reasoning engines might be manipulated to reach incorrect or harmful conclusions through carefully crafted inputs designed to exploit their logic.

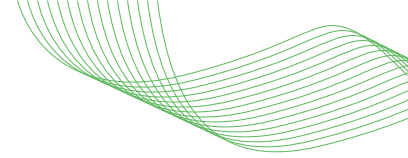
Agent orchestration systems could be compromised to bypass security controls, potentially allowing unauthorized access to protected functionality. Organizations must maintain comprehensive visibility into all components that extend model capabilities and ensure appropriate security controls are implemented at each integration point.

## Supply Chain Documentation Requirements

When deploying AI applications, comprehensive supply chain metadata must document the entire ecosystem of components to enable effective security management. This documentation should include all filtering components with version information and configuration parameters, along with RAG infrastructure details including embedding models and retrieval algorithms. Vector database providers and isolation mechanisms should be clearly specified, as should serving infrastructure components and their security configurations.

Organizations should also document caching systems with isolation and encryption specifications and all tools, plugins, and external systems accessible during inference. Authentication and authorization mechanisms need thorough documentation, including trust relationships and privilege boundaries. Dependency trees for all components with integrity verification methods should complete the documentation to ensure a comprehensive view of the supply chain.

This documentation enables security teams to conduct comprehensive risk assessments, implement appropriate controls, and respond effectively to vulnerabilities discovered in any component of the complex AI application stack. Regular updates to this documentation are essential as components evolve and new dependencies are introduced over time.



## 2.3 Scope of Existing Secure AI Frameworks

The rapid evolution of AI technologies has prompted the development of several industry frameworks designed to address security threats specific to AI systems. While these frameworks provide valuable guidance for securing aspects of AI development and deployment, they generally lack comprehensive coverage of AI supply chain security concerns.

Three prominent frameworks have emerged as reference points for organizations implementing AI security:

1. Google's Secure AI Framework (SAIF) provides a structured approach to AI security with emphasis on development practices and known threat mitigation.
2. MITRE's Adversarial Threat Landscape for Artificial-Intelligence Systems (ATLAS) extends the traditional ATT&CK framework to specifically address adversarial machine learning threats.
3. Microsoft's AI Security Bug Bar offers a taxonomy of AI-specific security issues with corresponding mitigation strategies.

Additionally, the OWASP AI Exchange has developed resources focusing on generative AI threats, particularly for Large Language Models.

While these frameworks make significant contributions to AI security, they share common limitations when addressing supply chain concerns. They typically focus on threats to deployed models rather than vulnerabilities introduced throughout the development pipeline. Critical gaps include insufficient attention to model provenance, cryptographic integrity verification, and risks associated with pre-trained models from untrusted sources.

As organizations increasingly build systems on open-source components, cloud APIs, and foundation models, comprehensive supply chain security becomes essential for trustworthy AI deployment. More material on these and other frameworks is available at [PROVIDE REFERENCE TO REPOSITORY OF ARTIFACTS].

## 3. Risks, Threats, and Existing Mitigations

### 3.1 Model Generation

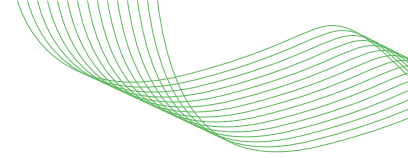
#### 3.1.1 Data Poisoning: Threats and Mitigations in AI Supply Chains

##### Understanding Data Poisoning Threats

Data poisoning represents a sophisticated attack vector that poses significant risks to AI model security and reliability. This threat involves the intentional manipulation of training or fine-tuning data to compromise a model's intended behavior and outputs. Even small, targeted modifications to training datasets can produce substantial downstream consequences, often remaining undetected until model deployment when erroneous predictions begin to surface.

The vulnerability to poisoning extends throughout the AI development lifecycle, with multiple potential entry points for adversaries. During data collection and preprocessing, attackers might directly target data sources by injecting malicious samples or subtly altering existing data. For example, in image classification systems, poisoned data might include subtly modified images with incorrect labels, causing the model to learn faulty associations. Training datasets containing URLs that reference external, uncontrolled data sources are particularly vulnerable to modification, either malicious or accidental.

Natural language processing models face similar risks, where carefully crafted text sequences can introduce bias or trigger specific behaviors. In some cases, poisoned data may even introduce backdoors that remain dormant until activated by specific input patterns or sequences, creating serious security vulnerabilities that bypass traditional testing.



### Indirect Poisoning Mechanisms

Beyond direct manipulation, data poisoning can occur through indirect means. Adversaries may post malicious or incorrect information on public internet sources, anticipating that AI crawlers will eventually collect this data for training purposes. This strategy has been observed both as an attack vector and as a defensive technique employed by content creators attempting to prevent AI systems from training on their copyrighted materials.

Post-ingestion stages introduce additional vulnerabilities. During data cleaning processes—which include deduplication, error correction, missing data imputation, bias correction, and labeling—human reviewers might accidentally or intentionally introduce errors. At scale, automated data curation tools might contain bugs or vulnerabilities that adversaries could exploit to manipulate data systematically.

### Training-Time Vulnerabilities

Even during the training process itself, data poisoning risks persist. Modern training methodologies typically assign different weights to various data sources, giving higher importance to trusted sources compared to general internet content. This approach helps ensure that authoritative information receives greater emphasis than potentially misleading content. However, these mixture weights significantly influence training outcomes and must be thoroughly documented to maintain model provenance.

The manipulation of these mixture weights represents an additional attack vector. An adversary with access to training infrastructure could alter weight configurations to amplify the influence of compromised data sources while diminishing trusted sources. Without proper monitoring and documentation of these parameters, such manipulations might go undetected.

### Storage Security Considerations

Direct data alteration while in storage presents another significant risk, both before and during training. If security controls fail to adequately restrict storage access, adversaries could bypass application-level protections and directly overwrite stored data. This emphasizes the critical importance of data integrity protections, comprehensive access controls, and thorough logging of all write operations throughout the AI development infrastructure.

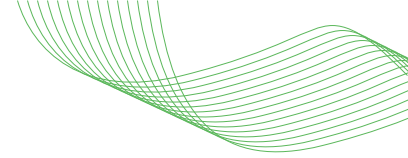
### Mitigation Through Data Provenance

Among the most effective defenses against data poisoning is the implementation of robust data provenance practices. Data provenance encompasses comprehensive documentation and tracking throughout the data lifecycle, from original sources through every transformation and processing step.

Effective provenance strategies require maintaining detailed records of all data sources, including information about origins, collection methodologies, and initial preprocessing steps. For computer vision applications, this includes tracking whether images originated from public datasets, web scraping, or proprietary collections. As data undergoes transformations during preprocessing, cleaning, and augmentation, each step must be logged with specific operations, parameters, and tooling details.

Version control for data, similar to software versioning practices, provides additional protection. By establishing comprehensive tracking of data sources and transformations, organizations can more readily identify pipeline issues and potential compromise. Regular source monitoring and updates further enhance detection and prevention capabilities.

### Threat Summary and Mitigations



Threat	Description	Mitigation	Impact
Inadequate Data	Using irrelevant data for model fine-tuning	Comprehensive data provenance with mixture weight documentation	Reduced training efficiency, model inaccuracy
Unauthorized Data	Training on copyrighted materials or using user data without consent	Complete data provenance with source verification	Legal and regulatory consequences
Data Poisoning	Deliberate modification of training data	Data provenance, access controls, integrity verification	Model misbehavior, security vulnerabilities
Invalid Transformations	Data corruption during cleaning processes	Complete lineage tracking of data transformations	Model misbehavior, unpredictable outputs

### 3.1.2 Model Training Security: Risks and Mitigations

#### Components of Model Training

Model training encompasses three critical components that require security consideration: the training data (covered in the previous section), the model architecture code (including architecture definitions and hyperparameters), and the framework and libraries used in the training process. An additional component often involved is pre-trained models used as starting points for adaptation through fine-tuning or quantization.

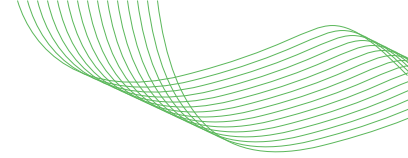
#### Open-Source Dependencies and Security Implications

The AI development ecosystem relies heavily on open-source libraries and third-party dependencies, offering developers significant advantages in functionality and efficiency. However, this reliance introduces distinct security challenges that organizations must address through systematic risk management.

One primary concern involves vulnerable or outdated dependencies. Open-source libraries, despite their utility, may contain both known and unknown security vulnerabilities. Outdated dependencies frequently lack essential security patches, making them susceptible to documented attack vectors. Understanding this vulnerability, attackers strategically target widely-used libraries and those with known security flaws.

Consider a scenario where researchers identify a critical vulnerability in a popular machine learning framework. Models built using affected versions become potential targets, with attackers leveraging the vulnerability to compromise model integrity or the underlying serving infrastructure. This risk intensifies when organizations face challenges updating dependencies due to compatibility issues, breaking changes, or resource constraints for proper testing after updates.

Supply chain attacks targeting dependencies of machine learning frameworks present another significant risk vector. Machine learning practitioners cannot reasonably track versions of all transitive dependencies required by their frameworks. Instead, organizations should implement supply chain metadata tools such as Software Bill of Materials (SBOM) and Supply-chain Levels for Software Artifacts (SLSA), aggregated through systems like Graph for Understanding Artifact Composition (GUAC). This approach allows ML practitioners to focus on model development while automated workflows verify dependency security, checking for recent versions, known vulnerabilities, and supply chain incidents.



### Model Architecture Code Protection

The code defining model architecture requires robust protection measures. Parameters such as layer configurations, activation functions, transformer counts, and attention head specifications significantly impact model performance. More concerning, malicious actors could modify architecture code to create backdoors that activate only with specific inputs, executing unauthorized operations.

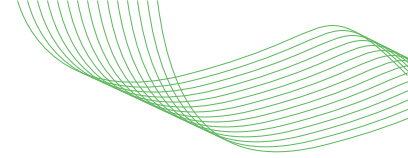
Detecting such backdoors requires thorough source code analysis, making proper code review essential before initiating training pipelines. Systems that allow developers to import local, un-reviewed changes before production-grade model training create significant insider threat risks. Organizations must implement rigorous source control security to prevent compromise of model architecture code.

### Pre-trained Model Integrity

When incorporating pre-trained models, organizations face the risk of tampering. Training pipelines must verify the integrity of all inputs, including software artifacts, datasets, and pre-trained models. Similar concerns apply to storage used for training checkpoints during extended training runs. A malicious insider could alter recently written checkpoints and force pipeline restarts, effectively injecting compromised models in ways that traditional detection methods might miss.

### Threat Summary and Mitigations

Threat	Description	Mitigation	Impact
Vulnerable Dependency	Training pipeline dependencies containing security vulnerabilities	Comprehensive supply chain visibility, regular dependency updates, dependency documentation for each training run	Vulnerabilities may enable adversaries to manipulate model behavior through specific inputs
ML Framework Vulnerability	Security flaws in the core ML framework used for training or inference	Regular framework updates, version documentation in supply chain records	Vulnerabilities in components like text parsing could compromise model serving systems or training infrastructure
Model Source Tampering	Unauthorized modification of model architecture code to introduce backdoors	Two-person code review requirements, secure source repositories with access controls	Insider threats could introduce architectural backdoors triggering specific behaviors
Model Weight Tampering	Unauthorized modification of model weights after training, during checkpoints, or in pre-trained models	Access controls for model storage, cryptographic model signatures	Weight manipulation can introduce model misbehavior under specific conditions
Compromised Evaluation	Non-reproducible evaluation processes allowing fraudulent results to promote malicious models	Tamper-proof documentation of evaluation protocols and results	Compromised models may exhibit unexpected behavior affecting specific user categories



### 3.1.3 Application Integration in the AI Supply Chain

#### The Critical Application Layer

Models by themselves provide limited value until integrated into applications that enable user interaction, data processing, and decision implementation. The application layer thus represents a crucial element of the AI supply chain, introducing additional security considerations that organizations must address when developing or consuming AI-powered services.

#### Application Dependency Supply Chain

The application supply chain encompasses numerous components that directly impact security posture, including:

1. Open-source libraries and frameworks used in application development
2. Third-party services and APIs integrated into the application architecture
3. Additional data sources and retrieval mechanisms (such as RAG implementations)
4. Client-side components and user interfaces interacting with models

These components mirror many traditional software supply chain concerns while introducing AI-specific considerations. Application dependencies often have privileges to access model inputs and outputs, potentially enabling manipulation of critical data flows if compromised. This privileged position makes application security particularly consequential in the overall AI supply chain.

Effective dependency management constitutes a foundational element of supply chain risk mitigation. Organizations must implement systematic approaches to tracking, validating, and updating all application dependencies. This includes maintaining comprehensive Software Bills of Materials (SBOMs), implementing automated vulnerability scanning in CI/CD pipelines, and establishing clear update policies to address emerging security threats promptly.

#### Evaluating Dependency Security

When selecting dependencies for AI applications, organizations should conduct thorough supply chain security assessments. Critical factors include:

Security track record of the dependency maintainers, including response time to vulnerability reports and transparency in security communications. Compromise of a widely-used library can simultaneously affect thousands of downstream AI applications, making maintainer security practices particularly significant.

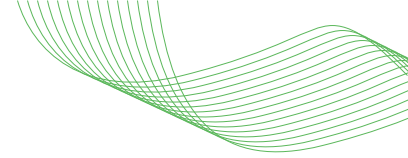
Development activity and community engagement levels serve as indicators of security attention. Abandoned or infrequently maintained dependencies are less likely to receive timely security updates, creating potential vulnerability gaps in the application supply chain.

Dependency licensing and provenance verification ensure legal compliance while reducing the risk of malicious code insertion. Organizations should implement automated checks for license compatibility and cryptographic verification of package integrity to prevent supply chain attacks through compromised dependencies.

#### Format-Specific Supply Chain Risks

Certain data formats and serialization methods introduce specific supply chain vulnerabilities. Most notably, formats like Python's pickle module enable arbitrary code execution during deserialization, creating significant security risks if used within applications or for model persistence.

Organizations should implement strict policies prohibiting unsafe serialization formats throughout the application supply chain. Secure alternatives such as JSON, Protocol Buffers, or format-specific ML frameworks (ONNX, SavedModel) provide similar functionality without the associated



security risks. When consuming third-party AI services, organizations should verify which serialization formats are used throughout the application stack.

### Communication Channel Security

When models are not collocated with applications—as in many cloud-based or microservice architectures—the communication channels between application components introduce additional supply chain considerations. Transit security becomes essential for preserving both data confidentiality and model integrity.

Organizations should implement end-to-end encryption for all model-application communications, with appropriate certificate validation and pinning to prevent man-in-the-middle attacks. API authentication mechanisms should enforce least-privilege principles, limiting access to only the specific model capabilities each application component requires.

For high-security environments, organizations should consider implementing additional integrity verification measures such as signed model responses or zero-knowledge proofs to detect supply chain compromises at the communication layer.

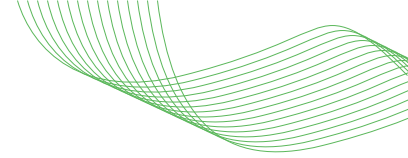
### Threat Summary for Application Supply Chain

Supply Chain Threat	Description	Mitigation Strategies	Impact
Compromised Dependencies	Malicious code insertion or vulnerability exploitation in application libraries	Comprehensive dependency validation, SBOMs, automated scanning, verified builds	Data exfiltration, model manipulation, unauthorized access
Unsafe Serialization Formats	Use of formats enabling code execution during deserialization	Format policy enforcement, secure alternatives, automated scanning	Remote code execution, application compromise
Vulnerable Communication Channels	Interception or manipulation of model-application traffic	End-to-end encryption, certificate validation, response signing	Data exposure, model input/output manipulation
Abandoned Dependencies	Security vulnerabilities in unmaintained application components	Dependency health monitoring, lifecycle policies, replacement strategies	Persistent unpatched vulnerabilities, degraded security posture
Third-Party API Compromise	Security failures in external services integrated into the application	API security assessments, traffic monitoring, fallback mechanisms	Supply chain infiltration, data leakage, service disruption

## 3.1.4 AI Infrastructure Supply Chain Security

### Third-Party Infrastructure Risks in the AI Supply Chain

When consuming AI models hosted on third-party infrastructure, organizations face unique supply chain security challenges that extend beyond traditional software concerns. The infrastructure layer—including cloud platforms, containerization services, and orchestration systems—constitutes a critical component of the AI supply chain with distinct security implications for model consumers.



## Evaluating Infrastructure Provenance

Model consumers must consider the complete provenance of AI infrastructure components when assessing supply chain security. This includes understanding not only where a model was trained but also the security posture of the infrastructure hosting the model for inference. Third-party providers may utilize complex service chains with numerous dependencies, creating a multi-tiered supply chain where vulnerabilities at any level could compromise model integrity or confidentiality.

Organizations should request comprehensive documentation of infrastructure components, including cloud service providers, container images, orchestration systems, and network configurations. Providers committed to supply chain security will maintain detailed records of infrastructure-as-code deployments, container image signatures, and dependency verification processes that consumers can audit.

## Model Serving Infrastructure Vulnerabilities

The model serving infrastructure represents a particularly sensitive element in the AI supply chain. When consuming models hosted by third parties, organizations effectively extend their trust boundary to include external serving systems that may process sensitive data or make critical decisions.

Key concerns include:

1. **Data Transit Exposure:** User data transmitted to third-party inference endpoints may be intercepted, modified, or analyzed during transit, potentially compromising confidentiality or introducing adversarial inputs.
2. **Multi-Tenant Isolation Failures:** Most AI serving infrastructure operates in multi-tenant environments where inadequate isolation between customers could lead to data leakage or side-channel attacks that compromise model behavior.
3. **Infrastructure Configuration Drift:** Third-party providers may modify serving infrastructure configurations without notice, potentially introducing vulnerabilities or changing model behavior in subtle ways that downstream consumers cannot detect.
4. **Dependency Supply Chain Compromises:** Serving infrastructure typically depends on numerous software components (web servers, API gateways, monitoring systems) that represent additional attack surfaces if compromised through supply chain attacks.

## Development Environment Supply Chain Risks

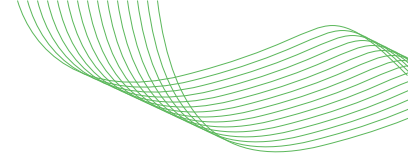
When consuming models, organizations should consider the development environments where those models originated. Notebook environments and development tools constitute a frequently overlooked yet critical infrastructure component in the AI supply chain.

Models developed in compromised notebook environments may contain subtle backdoors or vulnerabilities introduced through poisoned dependencies, compromised development tools, or malicious code execution. These vulnerabilities can persist through the entire supply chain, eventually affecting consumers of the hosted models.

Third-party model providers should implement and document security controls for development environments, including version-controlled notebooks, dependency scanning, privileged access management, and integrity verification of development artifacts. Model consumers should request evidence of these controls as part of supply chain security assessment.

## Training Pipeline Supply Chain Integrity

The security of the training pipeline directly impacts the trustworthiness of resulting models. When consuming hosted models, organizations must evaluate whether appropriate supply chain controls were implemented during the training process.



Specific concerns include:

1. **Checkpoint Integrity:** During extended training runs, model checkpoints may be saved and loaded multiple times. Without cryptographic verification, compromised checkpoints could introduce backdoors or bias into the final model.
2. **Training Job Resource Integrity:** Vulnerable or compromised training infrastructure could allow attackers to manipulate model weights or introduce backdoors during the training process, creating vulnerabilities that persist in the hosted model.
3. **Model Conversion Supply Chain:** Format conversion processes between different serialization formats represent additional attack surfaces in the AI supply chain. Unsafe serialization formats like pickle can introduce code execution vulnerabilities that compromise both the model and the infrastructure hosting it.

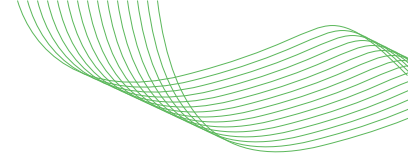
### Supply Chain Assurance for Infrastructure

When consuming models from third-party infrastructure, organizations should request evidence of supply chain security measures including:

1. **Infrastructure Attestation:** Cryptographic proof that infrastructure deployments match declared configurations and have not been tampered.
2. **Dependency Scanning Reports:** Documentation showing that infrastructure components have been scanned for vulnerabilities throughout the development lifecycle.
3. **Secure Access Logs:** Evidence of appropriate access controls and monitoring for infrastructure components with alerts for anomalous activity.
4. **Model Provenance Verification:** End-to-end tracking of model artifacts from development through deployment, with cryptographic signatures ensuring integrity at each stage.
5. **Third-Party Security Assessments:** Independent validation of infrastructure security controls by trusted external auditors.

### Threat Summary for Infrastructure Supply Chain

Supply Chain Threat	Description	Consumer Mitigation Strategies	Impact
Opaque Infrastructure Provenance	Insufficient visibility into hosting infrastructure components and their security posture	Request detailed infrastructure documentation, attestations, and third-party audit results	Unknown vulnerabilities, inability to perform accurate risk assessment
Serving Infrastructure Compromise	Vulnerabilities in model serving platforms that may affect multiple hosted models	Evaluate isolation mechanisms, request penetration testing reports, implement client-side verification	Potential data exposure, model manipulation, or adversarial attacks
Development Environment Artifacts	Backdoors or vulnerabilities introduced during model development persisting into production	Request development security documentation, perform independent model validation and testing	Unexpected model behavior, data leakage, or security bypass



Supply Chain Threat	Description	Consumer Mitigation Strategies	Impact
Checkpoint and Training Integrity	Manipulation of model training processes affecting the integrity of the final model	Require cryptographic verification of training artifacts, reproducible training processes	Model backdoors, bias, or performance degradation
Third-Party Dependency Compromise	Supply chain attacks targeting dependencies of the hosting infrastructure	Review dependency management practices, request SBOM for infrastructure components	Potential remote code execution, privilege escalation, or lateral movement

### 3.2 Model Integration and Consumption

Model Integration and Consumption encompasses the critical phase where organizations incorporate previously generated models into their systems, typically followed by fine-tuning prior to deployment. This stage presents several significant security challenges originating from multiple vectors: the operational data processed by the live system, the third-party models being integrated, the application stack and the underlying infrastructure itself. Each of these components introduces distinct vulnerabilities that must be systematically addressed to maintain supply chain integrity throughout the integration process.

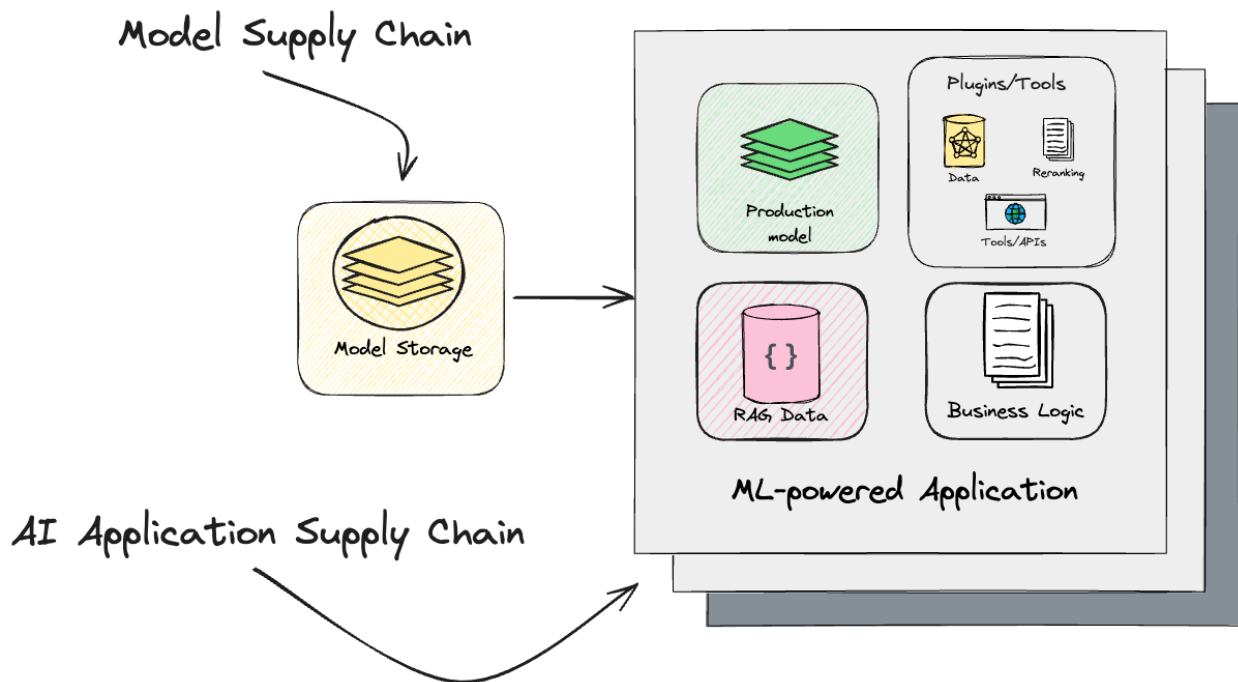
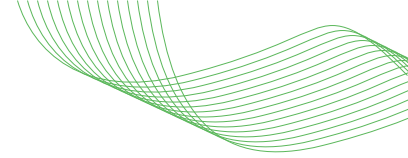


Figure 4: integration

#### 3.2.1 Supply Chain Security for Data



### Data Consumption and Processing

AI-powered systems fundamentally transform traditional information processing by efficiently handling unstructured data that conventional systems cannot readily process. These systems extract valuable insights through advanced processing techniques, creating novel security considerations throughout the supply chain.

AI systems ingest data through multiple vectors:

- Continuously updated external systems (e.g., internet sources)
- Traditional database query mechanisms
- Raw or preprocessed data repositories accessed via embedding vector-based unstructured queries
- Fine-tuning processes that incorporate integrator/consumer data for specialized tasks, often leveraging efficient techniques like LoRA (Low-Rank Adaptation)

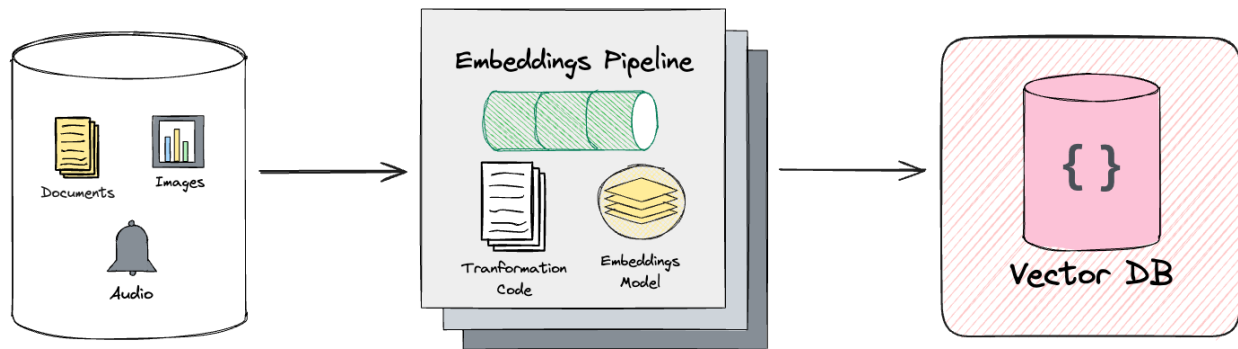


Figure 5: embeddings

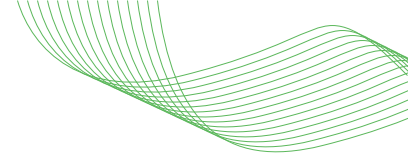
### Security Implications

Data security and integrity demands rigorous attention to user-data interactions and internal data processing workflows. Threat actors can exploit vulnerabilities across the data flow, potentially compromising model behavior, extracting sensitive information, or undermining system integrity. Proper threat modeling must account for these specialized attack vectors.

While encryption represents a potential mitigation for data leakage, AI models generally require unencrypted data for operation. This necessitates carefully defined trust boundaries to prevent unauthorized exposure of unencrypted information across security domains.

### Threat Matrix and Mitigations

Threat	Description	Mitigation	Impact
<b>Data Leakage (output)</b>	Unintended exposure of sensitive information via model outputs	<ul style="list-style-type: none"> <li>· Implement robust output filtering mechanisms</li> <li>· Regularly audit and monitor model outputs</li> <li>· Establish rate limiting on similar queries</li> </ul>	Exposure of confidential data, PII, or intellectual property

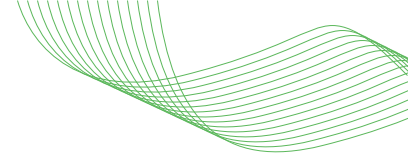


Threat	Description	Mitigation	Impact
<b>Data Poisoning</b>	Manipulation of external data sources leading to compromised model behavior	<ul style="list-style-type: none"> <li>· Implement data validation pipelines</li> <li>· Establish data provenance tracking</li> <li>· Create anomaly detection for incoming data</li> </ul>	Degraded model performance, security vulnerabilities, or backdoors
<b>Feedback Data Privacy</b>	User feedback containing PII or other sensitive information	<ul style="list-style-type: none"> <li>· Implement strict data anonymization techniques</li> <li>· Use secure, encrypted storage for feedback data</li> <li>· Establish retention policies</li> </ul>	Privacy violations and potential legal/regulatory consequences
<b>Cross RAG Contamination</b>	Information leakage between different RAG instances or unauthorized data mixing	<ul style="list-style-type: none"> <li>· Enforce strict data isolation</li> <li>· Implement comprehensive data lineage tracking</li> <li>· Conduct regular security boundary audits</li> </ul>	Data privacy violations and security boundary breaches
<b>Vector DB Injection</b>	Manipulation of embedding vectors or metadata affecting retrieval results	<ul style="list-style-type: none"> <li>· Validate inputs rigorously</li> <li>· Implement embedding sanitization</li> <li>· Enforce access controls on vector operations</li> </ul>	Compromised retrieval results and potential security vulnerabilities
<b>Vector Space Attacks</b>	Exploiting similarities in embedding space to manipulate system behavior	<ul style="list-style-type: none"> <li>· Monitor embedding space for anomalies</li> <li>· Implement adversarial testing</li> <li>· Apply dimensional security controls</li> </ul>	Unauthorized data access or manipulation of system behavior
<b>Embeddings Confusion</b>	Representation conflicts from different models, low precision, or third-party services	<ul style="list-style-type: none"> <li>· Use trusted embedding sources and models</li> <li>· Standardize embedding generation</li> <li>· Enhance embeddings robustness and precision</li> </ul>	Information retrieval failures, security bypasses (LLM08:2025)

### Recommended Security Controls

To mitigate these threats effectively, organizations should implement a defense-in-depth strategy that includes:

1. Comprehensive data provenance tracking throughout the AI supply chain



2. Regular security assessments focused on AI-specific attack vectors
3. Robust input/output filtering with continuous monitoring
4. Clear separation of security domains with defined trust boundaries
5. Standardized embedding generation processes with security validation

### 3.2.2 Model

#### Model Integration Fundamentals

The cornerstone of AI-powered systems is the model or collection of models integrated into the operational architecture. While models are not directly executed as traditional programs or processes, they significantly impact system confidentiality, integrity, and availability through specialized mechanisms that require dedicated security controls.

#### Internal Model Security Considerations

A common security misconception is that internally developed and stored models are inherently secure. This dangerous assumption overlooks critical vulnerabilities in the AI supply chain. Internally developed models remain susceptible to data poisoning attacks throughout their lifecycle, with malicious alterations particularly difficult to detect due to their limited inspectability. Privileged insiders with storage access could modify models while concealing their actions, leaving few forensic traces. Even robust training infrastructure can be compromised in ways that affect model integrity without triggering standard detection mechanisms, creating persistent security risks that traditional controls may not address.

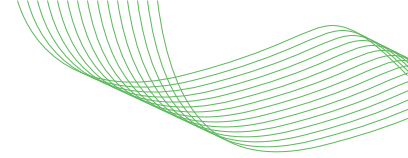
#### Cryptographic Assurance Mechanisms

Ensuring model integrity requires implementing robust cryptographic controls throughout the AI supply chain. Digital signatures serve as the primary mechanism for verifying model authenticity, with open source initiatives like Sigstore enabling comprehensive cryptographic signing of AI artifacts. Effective implementation requires signature verification at multiple stages: post-training validation, pre-deployment checks, and continuous runtime integrity verification. Hash verification provides an additional integrity layer when properly implemented, creating a multi-layered defense against unauthorized model modifications.

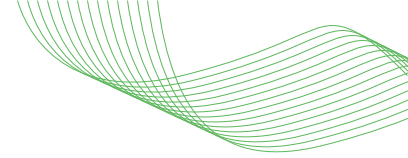
#### Supply Chain Visibility Framework

Effective incident response requires comprehensive visibility beyond basic tamper-resistance. Organizations must implement standardized recording of dependencies and build/training information to create auditable trails for security investigations. The SLSA framework establishes verifiable security posture metrics when applied to AI components, while AI-adapted Software Bills of Materials (SBOMs) enable rapid vulnerability identification and response. Tools like GUAC support continuous monitoring capabilities that can detect compromised models before deployment or fine-tuning, creating an essential early warning system for supply chain compromises.

#### AI Model Security Threats and Mitigations



Threat	Description	Mitigation	Impact
<b>Lack of Model Provenance or Incorrect Provenance</b>	The model's provenance information may be invalid, incomplete, or deliberately tampered with to conceal attacks. In some cases, provenance documentation may be entirely absent, creating significant security gaps.	Generate comprehensive model provenance using trusted build systems and ensure it exists in a tamper-proof format. Implement strict policies preventing the deployment of models lacking proper provenance documentation.	Invalid or corrupted models can enter production environments undetected. Security incident response becomes significantly more challenging without complete provenance records, extending remediation timelines.
<b>Data Drift</b>	Gradual changes in input data distribution over time that degrade model performance, or scenarios where external knowledge bases become outdated relative to real-world conditions.	Implement continuous distribution monitoring paired with scheduled model retraining protocols to adapt to evolving data patterns. Establish baseline performance metrics and automated drift detection thresholds.	Decreased accuracy and relevance of the model, potentially impacting business operations, decision quality, and overall system reliability.
<b>Model Inversion</b>	Supply chain security risk where sensitive data provided during model generation or fine-tuning becomes vulnerable to extraction through inference attacks targeting the deployed model.	Apply differential privacy techniques during training, implement comprehensive output filtering for sensitive data patterns, and conduct regular privacy audits of model responses to detect potential data leakage.	Exposure of confidential information or personally identifiable data depending on the training data supply chain, potentially compromising model integrity and undermining user trust.



**Model Serialization Attack**

Security vulnerability where model weights are serialized in unsafe formats (e.g., pickle) that permit arbitrary code execution when loaded into memory.

Implement exclusively safe model serialization formats with proper input validation and apply cryptographic signatures to verify model integrity before loading.

Potential arbitrary code execution during model deserialization, creating compromise vectors for serving infrastructure. If used as an intermediary format, training infrastructure and format conversion systems face similar risks.

**Recommended Implementation Strategy**

Organizations should implement end-to-end cryptographic verification for all model artifacts while establishing automated provenance verification for both models and their training data. This approach must integrate human-readable and machine-readable supply chain documentation to support both manual reviews and automated tools. Continuous integrity monitoring throughout the model lifecycle creates a foundation for effective security, supported by incident response playbooks specific to model tampering scenarios and least-privilege controls for all model storage and deployment systems.

This comprehensive approach ensures that organizations can effectively mitigate the unique supply chain risks associated with AI models while maintaining operational efficiency and security posture appropriate to their threat model.

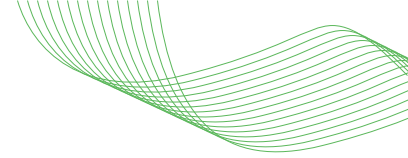
**3.2.3 Application**

Third-party AI applications introduce distinct supply chain security considerations requiring comprehensive risk assessment throughout the integration lifecycle. These applications present multifaceted security challenges spanning data protection, model integrity verification, and operational security controls that must be rigorously evaluated before deployment into production environments.

Data security represents a primary concern when integrating external AI applications. These systems typically require privileged access to sensitive organizational data, necessitating thorough evaluation of the application's data handling architecture. Organizations must conduct comprehensive assessments of data storage mechanisms, transmission security controls, and processing methodologies to verify compliance with relevant regulatory frameworks. This evaluation should examine encryption implementations, access control models, and data lineage tracking to establish sufficient protection against unauthorized access or exfiltration.

Model integrity verification constitutes a critical aspect of third-party application security assessment. Organizations must establish validation procedures to verify model provenance, development methodology, and testing protocols. This process includes detailed examination of training datasets, algorithm selection criteria, and potential bias factors that could impact model reliability. Comprehensive validation testing should be performed to confirm model accuracy, reliability, and absence of malicious code or backdoor functionality before production deployment.

The prevalent use of third-party code artifacts, particularly scripts and Jupyter notebooks, introduces additional supply chain risks requiring specialized controls. Organizations frequently leverage these components for model development and feature testing, often without comprehensive security validation. Such artifacts may contain malicious code segments designed to exe-



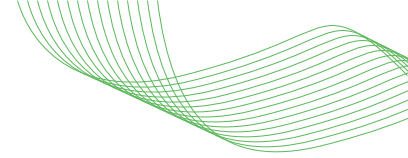
ecute unauthorized functions or exfiltrate sensitive information. Implementing rigorous code review processes and secure execution environments provides essential protection against these threats.

The extensive dependency on third-party MLOps tooling and development frameworks creates expanded attack surfaces requiring continuous monitoring. Each component in the development and deployment pipeline represents a potential attack vector for adversaries targeting AI systems. Organizations should prioritize solutions offering robust security features and formal security attestations. Maintaining comprehensive vulnerability management processes with specific focus on published CVEs affecting AI development components ensures timely remediation of security defects before exploitation.

Generative AI systems present unique risk profiles through potential control flow manipulation via indirect prompt injection techniques. These attacks can redirect model behavior without user awareness, potentially resulting in data exfiltration, unauthorized content generation, or execution of malicious actions on behalf of attackers. As these systems integrate with additional services and tools, their expanded capabilities introduce corresponding security considerations that require specialized detection and prevention mechanisms to maintain operational integrity.

### AI Supply Chain Security Threats

Threat	Description	Mitigation	Impact
<b>Indirect Prompt Injection</b>	Unauthorized instructions or malicious commands introduced to the model through external data sources during inference operations, including RAG systems, agentic frameworks, web search functionality, tool outputs, and other integrated data providers.	Implement comprehensive access control frameworks with strong authentication mechanisms for all external data sources. Deploy advanced content filtering systems with specific detection capabilities for injection patterns across all integrated data channels.	Potential exposure of confidential information or PII through manipulated model responses, execution of unauthorized actions through compromised model behavior, and undermining of trust boundaries between system components.
<b>Vulnerability in MLOps Tooling or ML Libraries</b>	Security defects present in development, deployment, or monitoring components within the AI infrastructure stack that can be exploited to compromise model integrity or operational security. These vulnerabilities often exist in underlying dependencies rather than primary components.	Implement systematic update management for all libraries and MLOps tooling with comprehensive vulnerability scanning. Document all tools and libraries in standardized supply chain artifacts for each training run to enable rapid impact assessment when vulnerabilities are discovered.	Exploitation of vulnerabilities in auxiliary systems such as text parsing components could enable attackers to compromise data augmentation pipelines, potentially introducing backdoors into models during training or deployment phases.



Threat	Description	Mitigation	Impact
<b>Insecure Function Calling</b>	Security weaknesses in the interface between language models and external function capabilities that can be exploited through control flow manipulation, allowing attackers to execute unauthorized actions when models are integrated with plugins, tools, or service APIs.	Restrict model-accessible functions to read-only operations where possible. Apply granular least-privilege principles to all model function access permissions. Implement role-based access control (RBAC) frameworks to limit model interaction capabilities based on authentication context.	Sophisticated attacks leveraging indirect prompt injection techniques could trigger undefined system behaviors with severe consequences, including unauthorized data access, privileged command execution, or destructive operations against connected systems.

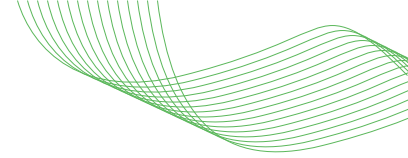
### 3.2.4 Infrastructure

When integrating and consuming AI models, the underlying infrastructure represents a critical component in the AI supply chain that directly influences system security and operational reliability. Infrastructure vulnerabilities constitute significant security vectors that can compromise the integrity, availability, and confidentiality of AI models and their processed data throughout the supply chain lifecycle. These risks extend beyond immediate operational concerns to impact model trustworthiness, inference quality, and compliance with evolving regulatory frameworks governing AI systems.

Infrastructure vulnerabilities manifest across the AI supply chain through multiple attack surfaces, including hardware component failures, software security defects, network communication disruptions, and inadequate security control implementation. Hardware failures within compute infrastructure can trigger availability interruptions that disrupt AI service delivery and model performance. Software vulnerabilities present particularly concerning supply chain risks, as they enable threat actors to potentially gain unauthorized access to sensitive training data or manipulate model outputs through integrity violations. Network disruptions across the AI pipeline can compromise data transmission integrity between system components, introducing both operational latency and security weaknesses.

The architectural complexity inherent in modern AI infrastructures significantly amplifies supply chain risk profiles. These environments typically leverage heterogeneous deployment models spanning on-premises computing resources and cloud-based infrastructure, creating expanded attack surfaces requiring comprehensive security controls. The proliferation of third-party dependencies throughout the AI supply chain—including external APIs, pre-trained model components, and managed services—introduces additional risk dimensions requiring thorough evaluation, as these components may operate under security practices misaligned with organizational requirements. A comprehensive understanding of infrastructure components and their security posture provides the essential foundation for effective AI supply chain risk management and vulnerability mitigation.

#### AI Infrastructure Security Threats

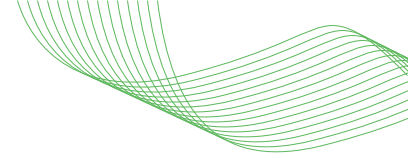


Threat	Description	Mitigation	Impact
<b>Cache Poisoning</b>	Malicious manipulation of cached data or results within the AI infrastructure, where attackers introduce corrupted information into temporary storage mechanisms used to optimize model performance.	Implement comprehensive cache validation protocols, cryptographic verification of cached content, and regular cache refresh cycles with integrity checks.	Degraded model performance, inconsistent outputs, potential exposure of sensitive information, and introduction of backdoors that persist across deployments.
<b>Feedback Loop Exploitation</b>	Adversarial manipulation of model training or reinforcement learning processes by introducing carefully crafted inputs designed to gradually shift model behavior in directions favorable to attackers.	Deploy anomaly detection for feedback patterns, implement robust validation gates for training data, and establish monitoring systems for unexpected model behavior drift.	Progressive degradation of model performance, introduction of systemic biases, potential data privacy violations, and loss of model integrity over extended periods.
<b>Overly Permissive Entitlements</b>	Excessive access rights within the AI infrastructure allowing components or users to access, modify, or control resources beyond operational requirements, creating unnecessary attack surfaces.	Implement comprehensive least-privilege architecture, regularly audit entitlement configurations, enforce just-in-time access protocols, and establish proper service account isolation.	Expanded attack surface for lateral movement, increased blast radius during security incidents, potential for privilege escalation, and undermined segmentation controls within AI systems.

### 3.2.4.1: Self-Hosted On-Premises Infrastructure

Self-hosted on-premises infrastructure represents a critical deployment model for AI systems, placing all components within an organization’s physical data centers under direct operational control. This approach provides comprehensive authority over the entire infrastructure stack—from hardware procurement to networking architecture, software deployment, and security control implementation. Organizations gain significant advantages through customization capabilities that can address specific technical requirements and regulatory compliance mandates. Industries with stringent data sovereignty requirements, particularly healthcare and financial services, often leverage this model to maintain complete control over sensitive data processing boundaries. However, this approach transfers complete responsibility for supply chain security verification to the implementing organization, requiring robust processes to validate the integrity of both hardware and software components throughout their lifecycle.

The self-hosted deployment model introduces substantial verification challenges requiring sophisticated technical controls. Organizations must implement comprehensive component validation processes including cryptographic verification through digital signatures, secure boot im-



plementation utilizing hardware root of trust (HrOT) mechanisms, and continuous integrity monitoring. These controls establish foundational trust in hardware components and software packages, providing protection against supply chain tampering during transit or installation. Secure boot processes specifically mitigate unauthorized code execution by cryptographically validating software authenticity during system initialization phases, creating substantial barriers to malware persistence.

Adherence to established security frameworks becomes essential for managing supply chain risk in self-hosted environments. Frameworks such as NIST Cybersecurity Framework and ISO/IEC 27001 provide structured methodologies for comprehensive risk management spanning the entire AI infrastructure supply chain. These frameworks emphasize continuous security monitoring, regular risk assessment processes, and formalized incident response capabilities. Implementation of these standards establishes systematic approaches to identifying, assessing, and mitigating vulnerabilities throughout the self-hosted AI supply chain ecosystem.

While self-hosted infrastructure delivers unmatched control and customization potential, it demands significant technical expertise and resource allocation. Organizations bear complete responsibility for defensive security posture, requiring implementation of defense-in-depth strategies including robust access controls, comprehensive network segmentation, data encryption, and advanced threat detection systems. Security control failures can result in unauthorized access to AI models and their underlying data, potentially triggering substantial breach scenarios and compliance violations. Organizations must carefully evaluate their capability to maintain effective security controls against the benefits of complete infrastructure ownership. All verified artifacts from the supply chain should be maintained in a secured Repository of Artifacts to ensure provenance tracking and auditability throughout the system lifecycle.

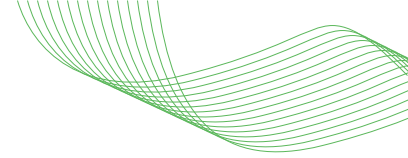
#### **3.2.4.2: Self-Hosted on Cloud or Third-Party Managed Infrastructure**

Self-hosted deployments on cloud or third-party managed infrastructure represent a strategic hybrid approach where organizations leverage external compute resources while maintaining operational control over critical AI deployment components. In this architecture, cloud providers assume responsibility for physical infrastructure management—including server hardware, storage systems, and network fabric—while organizations retain control over virtual machine configurations, operating system hardening, middleware security, API gateway implementation, and logical network segmentation.

This deployment model delivers significant operational advantages through resource elasticity, allowing organizations to dynamically scale computational capabilities based on fluctuating inference demands while optimizing cost structures through consumption-based pricing models. Cloud providers typically enhance the security baseline through integrated protective measures including automated threat detection mechanisms, advanced DDoS mitigation, and comprehensive perimeter controls that establish foundational security layers. These native capabilities can substantially strengthen the overall security posture of AI systems when properly configured and monitored.

The shared responsibility model introduces specialized security considerations that require precise delineation of security obligations between provider and customer. Organizations must implement rigorous configuration management practices across their virtual infrastructure components to prevent security vulnerabilities from emerging at architectural boundaries. Misconfigurations represent significant risk vectors, potentially enabling unauthorized access to model parameters, training data repositories, or inference endpoints. Improperly secured API gateways or inadequate network controls can create exploitation vectors for adversaries targeting model extraction or data exfiltration.

While offering substantial advantages in scalability and operational efficiency, this hybrid model necessitates careful management of trust relationships with infrastructure providers. Organiza-



tions must establish verification mechanisms to validate provider security claims while implementing continuous monitoring across the entire infrastructure stack to detect security anomalies. The effectiveness of this approach depends heavily on clearly defined security responsibilities, comprehensive visibility across the deployment architecture, and proper implementation of security controls at each layer of the technology stack. Organizations leveraging this model should maintain comprehensive security documentation within their Repository of Artifacts to ensure complete supply chain visibility and verifiable security posture.

### 3.2.4.3: Managed Services by Third-Party Providers

Managed services delivered by third-party providers represent a comprehensive outsourcing model where external vendors assume responsibility for AI model hosting, tuning, optimization, and operational management. This deployment architecture transfers significant infrastructure management responsibilities to specialized providers while organizations maintain limited control over critical security boundaries including network configurations, authentication mechanisms, authorization frameworks, and oversight of data transmission pathways.

This deployment approach leverages specialized expertise and optimized resource allocation from providers with established capabilities in AI operations management. These specialized service providers typically maintain advanced technical competencies in model optimization, deployment automation, and runtime management that can deliver superior performance metrics and operational efficiency. The delegation of infrastructure management responsibilities significantly reduces internal resource requirements, enabling organizations to reallocate technical resources toward core business objectives while leveraging provider-managed deployment platforms.

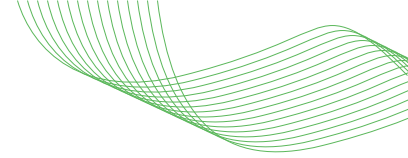
The managed service approach introduces distinct supply chain security considerations requiring robust governance frameworks. A primary concern involves potential visibility gaps across critical infrastructure components due to contractual boundaries with service providers. While organizations maintain theoretical control over security parameters, providers manage the majority of infrastructure components and model operations with varying transparency levels. This architecture presents significant challenges for regulatory compliance verification and security control validation. Organizations must implement comprehensive monitoring to ensure providers maintain stringent data protection controls and secure data transmission mechanisms throughout the service delivery pipeline. Provider security weaknesses can create direct exploitation vectors for sensitive data exposure or model integrity compromise if proper security validation mechanisms are not established.

Managed service provider relationships deliver substantial benefits through specialized expertise and operational efficiency but require rigorous risk assessment protocols. Organizations implementing this model must conduct comprehensive provider security assessments, establish detailed service level agreements with specific security requirements, maintain continuous monitoring capabilities, and implement verification mechanisms to ensure secure and compliant AI operations. Organizations should maintain comprehensive documentation of provider security certifications, audit reports, and control mappings within their Repository of Artifacts to ensure complete supply chain visibility and verifiable security posture across managed service boundaries.

## 4. Key Takeaways for Stakeholders in Your Organization

### 4.1 Executive Leadership

The accelerated integration of artificial intelligence technologies across enterprise environments has elevated critical supply chain security considerations requiring executive leadership attention. These multidimensional risks present significant implications for organizational operations, reputational integrity, regulatory compliance, and strategic positioning. As AI systems increase in



architectural complexity and autonomous capability, comprehensive supply chain security governance becomes an essential executive function requiring structured oversight frameworks.

The evolving nature of AI development introduces distinct security challenges throughout the technology lifecycle. AI technologies undergo continuous evolution with rapid emergence of novel algorithms, model architectures, and deployment patterns. This innovation velocity creates complex supply chain environments with specialized security considerations that traditional governance models may not adequately address. Executive leadership must implement structured evaluation protocols for system construction methodologies, validation frameworks, and integration architectures. Ensuring security boundary integrity, component compatibility, and operational reliability across diverse AI components requires executive-sponsored security programs to prevent cascading system failures and service disruptions.

Data integrity represents a foundational element of AI supply chain security requiring executive oversight. AI models derive their operational effectiveness directly from training data quality and integrity. Compromised, biased, or adversarially manipulated data introduces significant risks including algorithmic inaccuracies, discriminatory outcomes, and potential legal exposure. Leadership must establish comprehensive data governance frameworks addressing provenance verification, storage security, and usage authorization throughout the supply chain. This requires implementing robust data governance policies with specific AI considerations, ensuring encryption and access controls meet specialized requirements, and promoting ethical data collection methodologies within development organizations.

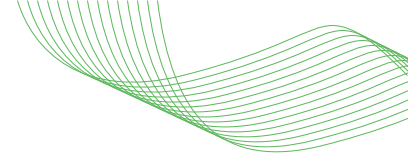
Supply chain disruption scenarios present substantial operational and financial implications requiring executive contingency planning. Leadership must develop comprehensive risk mitigation strategies addressing the unique characteristics of AI supply chains. This approach requires systematic identification of critical supply chain components, comprehensive dependency mapping across the AI architecture, and development of technical continuity plans addressing potential disruption scenarios. Proactive risk management enables organizations to maintain competitive positioning while establishing trust foundations for AI-powered products and services within increasingly regulated environments.

For executives seeking to optimize organizational AI supply chain security posture, the following strategic considerations should be prioritized:

1. Implement comprehensive discovery processes identifying AI utilization within internal systems and vendor services
2. Establish cross-functional governance teams with representation from security, legal, technical, and business stakeholders
3. Create dedicated ethics and risk oversight mechanisms focused specifically on AI supply chain security
4. Develop executive-level AI literacy programs ensuring leadership teams maintain sufficient technical understanding for effective governance

## 4.2 Security Practitioner

Security practitioners represent a critical component in the AI supply chain security ecosystem, implementing protective controls across the entire AI lifecycle from development through deployment and operational monitoring. While not typically involved in model architecture development, these professionals establish comprehensive security frameworks governing model development methodologies, validation requirements, and consumption patterns. Their specialized expertise ensures that security controls maintain effectiveness against evolving threat vectors throughout the AI supply chain.



Data security represents a foundational responsibility for practitioners engaged in AI security operations. These professionals develop and implement sophisticated protection strategies addressing data throughout its lifecycle—applying encryption for data at rest and in transit while ensuring alignment with evolving regulatory frameworks. This comprehensive approach includes implementing cryptographic controls, establishing granular access management frameworks, and deploying secure storage architectures with specific considerations for training data protection. These measures create essential safeguards against unauthorized access to sensitive training datasets and potential supply chain compromise through data poisoning vectors.

Model integrity verification constitutes a critical security function requiring specialized validation techniques. Security practitioners implement robust attestation methodologies that cryptographically verify model integrity and authenticity throughout the supply chain. These validation processes ensure models maintain integrity throughout development, fine-tuning, and deployment phases, creating verifiable trust relationships between model producers and consumers. This attestation framework establishes essential trust foundations for stakeholders throughout the AI ecosystem, enabling confident deployment of systems with verifiable security properties.

The operational security phase requires continuous vigilance through sophisticated monitoring frameworks. Security teams implement comprehensive surveillance systems identifying potential vulnerabilities and emerging threats across deployed AI infrastructure. Regular security assessment activities including architecture reviews and penetration testing exercises identify and remediate risks before exploitation. While many security practitioners may not directly participate in model development, their critical role in establishing defensive frameworks ensures operational reliability and security resilience as adoption of these systems accelerates across enterprise environments.

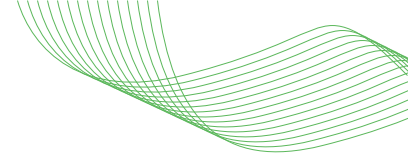
For security practitioners seeking to enhance organizational AI security posture, these strategic considerations warrant priority implementation:

1. Develop comprehensive security programs addressing model development and serving phases with specific focus on training pipeline security, fine-tuning integrity, and inference endpoint protection
2. Establish organization-specific risk acceptance frameworks addressing both open-source/open-weights models and commercial offerings with clear security requirements
3. Integrate AI-specific security considerations into existing supply chain security frameworks, addressing the unique technical and operational characteristics introduced by machine learning components

### 4.3 AI Researcher/Engineer

Research and development teams operate at the innovation frontier of artificial intelligence and machine learning technologies, continuously advancing algorithmic capabilities and architectural approaches. While focused primarily on technical innovation, these teams bear significant responsibility for implementing security considerations throughout the development lifecycle. AI researchers and engineers typically possess specialized expertise in algorithmic design, optimization techniques, and model architecture development, yet may lack comprehensive understanding of the complex security implications associated with their innovations. The security risk frameworks outlined across development phases provide essential guidance enabling research teams to implement security-by-design principles throughout the innovation process.

Security risk awareness constitutes a foundational capability for AI development organizations. By systematically identifying potential vulnerabilities across the development lifecycle, researchers can implement targeted controls addressing specific risk vectors. During initial data acquisition



phases, development teams with security awareness can establish rigorous validation methodologies ensuring dataset diversity, representativeness, and integrity while mitigating potential bias factors affecting model performance and fairness. This approach requires implementing structured data validation frameworks and comprehensive ethical guidelines governing data acquisition and processing activities throughout the research pipeline.

Cross-functional integration between research and security organizations represents a critical success factor for secure AI development. By establishing collaborative relationships with security specialists, development teams gain essential insights into emerging threat vectors and effective mitigation strategies. This structured collaboration cultivates security awareness within research organizations, transforming security from peripheral consideration to core development principle. The integration of security expertise throughout the research process enables early identification of potential vulnerabilities before they become embedded in deployed systems, significantly reducing remediation costs and implementation complexity.

For model developers and AI researchers integrating external models into organizational environments, these strategic considerations warrant priority implementation:

1. Recognize that AI architectures exist within complex supply chain ecosystems encompassing multiple interdependent components including data sources, infrastructure elements, application frameworks, and model artifacts—each introducing distinct security considerations requiring comprehensive security approaches
2. Implement standardized model documentation frameworks such as model cards capturing essential security characteristics, performance parameters, and operational limitations
3. Establish early engagement protocols with security organizations to identify potential risk vectors during initial development phases, enabling implementation of appropriate controls before architectural decisions become difficult to modify

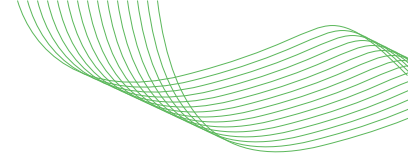
## 5. Conclusion

Artificial intelligence systems introduce distinctive security properties requiring specialized risk assessment methodologies beyond traditional application security frameworks. These systems possess unique characteristics including non-deterministic behavior patterns, complex data dependencies, and specialized attack surfaces that demand targeted security controls. As this technology domain matures, novel attack vectors targeting AI-specific vulnerabilities will continue to emerge, requiring continuous evolution of defensive capabilities. While addressing these specialized considerations, organizations must simultaneously maintain vigilance against conventional security threats including credential exposure within inference infrastructure and vulnerabilities within supporting development libraries and toolchains.

Comprehensive risk assessment requires systematic evaluation across the entire AI supply chain ecosystem comprising four interdependent components: Data acquisition and processing systems, Infrastructure deployment architecture, Application integration frameworks, and Model development methodologies. Effective security governance demands integrated threat modeling approaches incorporating both AI-specific attack vectors and traditional exploitation techniques within unified defense frameworks. This holistic security perspective enables development of targeted controls addressing the unique characteristics of machine learning systems while maintaining fundamental security principles throughout the technology stack.

This framework represents an evolving security methodology that will undergo continuous refinement as AI technologies and associated threat landscapes mature. Our Future work (and resulting technical publications) will address critical security domains including:

1. Cryptographic standards and implementation recommendations for model signing and



verification: Model Signing is a critical process in AI/ML development that involves cryptographically signing models during training to ensure their integrity and provenance. By embedding a digital signature, this initiative enables stakeholders to verify that the model in use is identical to the one originally developed, preventing tampering and IP theft. The benefits extend to strengthening the AI supply chain by providing traceability and ensuring models remain unaltered from creation to deployment. This enhances trust in AI systems, particularly in environments where different teams handle model training and production. By adopting Model Signing, organizations can safeguard their AI investments and maintain the reliability of their machine learning workflows. Our workstream plans on defining a standard workflow for how model developers sign their artifacts and how a consumer should securely consume a signed model within their environment.

2. Standardized model documentation frameworks with comprehensive interpretation guidelines for consumers: Model Cards are structured documentation designed to accompany machine learning models, providing critical insights into their intended use, performance metrics, limitations, and ethical considerations. This initiative aims to standardize components and metadata within Model Cards across the industry, addressing the current challenge of inconsistent formats and uneven informativeness in documents like Markdown files, PDFs, or webpages. By establishing a machine-readable format, the workstream seeks to enhance transparency, accountability, and trust in AI models. Additionally, integrating cryptographic signing for performance and security claims will ensure the integrity of the information provided. This effort will not only streamline model evaluation and governance but also empower stakeholders to make informed decisions by relying on consistent, verifiable, and comprehensive model documentation.

We also expect to review and discuss Supply chain Levels for Software Artifacts (SLSA) frameworks adapted specifically for model development pipelines.

Our technical community maintains continuous publication of security guidance through our GitHub repository at <https://github.com/cosai-oasis/wsl-supply-chain>. Organizations seeking to establish comprehensive AI security programs should integrate these risk identification frameworks with complementary defensive methodologies published through CoSAI's additional technical workstreams, enabling cohesive security strategies addressing the unique characteristics of AI supply chains.

Organizations interested in contributing to these technical security initiatives can engage through the Coalition for Secure AI at <https://www.coalitionforsecureai.org/join-us/>.

## 6. Contributors and Acknowledgements

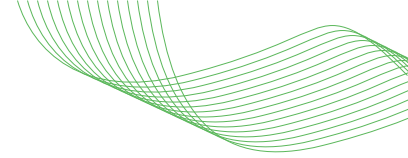
### Workstream Leads

- Matt Maloney, Cohere ([mattmaloney@cohere.com](mailto:mattmaloney@cohere.com))
- Andre Elizondo, Wiz ([andre.elizondo@wiz.io](mailto:andre.elizondo@wiz.io))
- Jay White, Microsoft ([jaywhite@microsoft.com](mailto:jaywhite@microsoft.com))

### Editors

- Mihai Maruseac, Google ([mihaimaruseac@google.com](mailto:mihaimaruseac@google.com))
- Eoin Wickens, HiddenLayer ([eoin@hiddenlayer.com](mailto:eoin@hiddenlayer.com))
- John Stone, Google ([thestone@google.com](mailto:thestone@google.com))

### Contributors



- Adam Tuaima, Trend Micro (adam\_tuaima@trendmicro.com)
- Arber Salihi, Thomson Reuters (Arber.Salihi@thomsonreuters.com)
- Danilo Tommasina, Thomson Reuters (danilo.tommasina@thomsonreuters.com)
- Daniel Rohrer, NVIDIA (drohrer@nvidia.com)
- Harish Thanneer, (harish.thanneer@intel.com), Intel
- Judith Furlong, Dell (Judith.Furlong@dell.com)
- Lorenzo Verstraeten, Thomson Reuters (lorenzo.verstraeten@thomsonreuters.com)
- Michael Katz, PayPal (mickatz@paypal.com)
- Ian Molloy, IBM (molloyim@us.ibm.com)
- Yassine Ilmi, Thomson Reuters (yassine.ilmi@thomsonreuters.com)
- Victor Lu (Victorjunlu@gmail.com)

#### **Technical Steering Committee Co-Chairs**

- Akila Srinivasan, Anthropic (akila@anthropic.com)
- J.R. Rao, IBM (jrrao@us.ibm.com)

#### **Reviewers**

- David Labianca, Google (ddlb@google.com)
- Omar Santos, Cisco (osantos@cisco.com)
- Sarah Novotny, GenLab (sarah@genlab.studio)

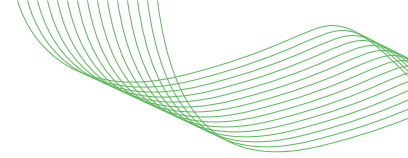
## **7. Appendix**

### **7.1 CoSAI Focus**

CoSAI is an OASIS Open Project, bringing together an open ecosystem of AI and security experts from industry-leading organizations. The project is dedicated to sharing best practices for secure AI deployment and collaborating on AI security research and product development. The scope of CoSAI is specifically focused on the secure building, integration, deployment, and operation of AI systems, with an emphasis on mitigating security risks unique to AI technologies. Other aspects of Trustworthy AI are deemed important but beyond the scope of the project including, ethics, fairness, explainability, bias detection, safety, consumer privacy, misinformation, hallucinations, deep fakes, or content safety concerns like hateful or abusive content, malware, or phishing generation. By concentrating on developing robust measures, best practices, and guidelines to safeguard AI systems against unauthorized access, tampering, or misuse, CoSAI aims to contribute to the responsible development and deployment of resilient, secure AI technologies.

### **7.2 Guidelines on usage of more advanced AI systems (e.g. large language models (LLMs), multi-modal language models. etc) for drafting documents for OASIS CoSAI:**

tl;dr: CoSAI contributions are actions performed by humans, who are responsible for the content of those contributions, based on their signed OASIS iCLA (and eCLA, if applicable). [Each contrib-



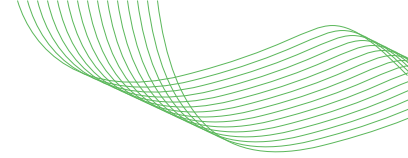
utor must confirm whether they are entitled to donate that material under the applicable open source license; OASIS and the CoSAI Project do not separately confirm that.] Each contributor is responsible for ensuring that all contributions comply with these AI use guidelines, including disclosure of any use of AI in contributions.

- Selection of AI systems: CoSAI recommends the use of reputable AI systems (lowering the risk of inadvertently incorporating infringing material).
- Model constraints: Currently, CoSAI or OASIS are not required to have a contract or financial agreement for using AI systems from specific vendors. However, CoSAI editors should consider employing varying tools to avoid potential fairness concerns among vendors.
- IP infringement: It is the responsibility of the individual who subscribes/prompts and receives a response from an AI system to confirm they have the right to repost and donate the content to OASIS under our rules.
- Transparency: CoSAI's goal will be to maintain transparency throughout the process by documenting substantial use of AI systems whenever possible (e.g., the prompts and the AI system used), and to ensure that all content, regardless of production by human or AI systems, was reviewed and edited by human experts. This helps build trust in the standards development process and ensures accountability.
- Human-edited content and quality control: CoSAI mandates human-reviewed or -edited results for any final outputs. A robust quality control process should be in place, involving careful review of the generated content for accuracy, relevance, and alignment with CoSAI's goals and principles. Human experts should scrutinize the output of AI systems to identify any errors, inconsistencies, or potential biases.
- Iterative refinement: The use of AI systems in drafting standards should be seen as an iterative process, with the generated content serving as a starting point for further refinement and improvement by human experts. Multiple rounds of review and editing may be necessary to ensure the final standards meet the required quality and reliability thresholds.

### 7.3 Copyright Notice

Copyright © OASIS Open 2025. All Rights Reserved. This document has been produced under the process and license terms stated in the OASIS Open Project rules: <https://www.oasis-open.org/policies-guidelines/open-projects-process>.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns. This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OASIS AND ITS MEMBERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THIS DOCUMENT OR ANY PART THEREOF. The name "OASIS" is a trademark of OASIS, the owner and developer of this document, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, documents, while reserving the right to enforce its marks against misleading uses. Please see



<https://www.oasis-open.org/policies-guidelines/trademark/> for above guidance.

This is a Non-Standards Track Work Product. The patent provisions of the OASIS IPR Policy do not apply.