Pharos Series: Report No. 5 July 2026

Navigating Security in the Machine Learning and AI Supply Chain: What Policymakers Need to Know

Max Smeets, Anna Sophie den Ouden, James Shires

virtual routes

PHAROS SERIES
Virtual Routes | www.virtual-routes.org
Design & Layout by Frank Wo | Cover by Vahram Muradyan

Pharos Series: Report No. 5 July 2026

Navigating Security in the Machine Learning and AI Supply Chain: What Policymakers Need to Know

Max Smeets, Anna Sophie den Ouden, James Shires

virtual routes

About the Authors

Max Smeets

Max Smeets is the Co-Director of Virtual Routes and serves as Managing Editor of *Binding Hook*. He also holds research positions at ETH Zurich, the Royal United Services Institute (RUSI), and Stanford University's Center for International Security and Cooperation. Max is the author of *Ransom War: How Cyber Crime Became a Threat to National Security* and *No Shortcuts: Why States Struggle to Develop a Military Cyber Force*.

Max received a BA in Economics, Politics, and Statistics from University College Roosevelt, Utrecht University, and an MPhil (Brasenose College) and DPhil (St. John's College) in International Relations from the University of Oxford.

Anna Sophie den Ouden

Anna Sophie den Ouden is a Junior Researcher at Virtual Routes and a Junior Lecturer at Leiden University's Institute of Security and Global Affairs. Her research primarily focuses on AI security, the impact of AI on the social and military domains, cyber conflict, and the geoeconomics of AI infrastructures. She also conducts research and leads data collection efforts for Action on Armed Violence. Prior positions include her roles at the Hague Centre for Strategic Studies and the United Nations Project Office on Governance.

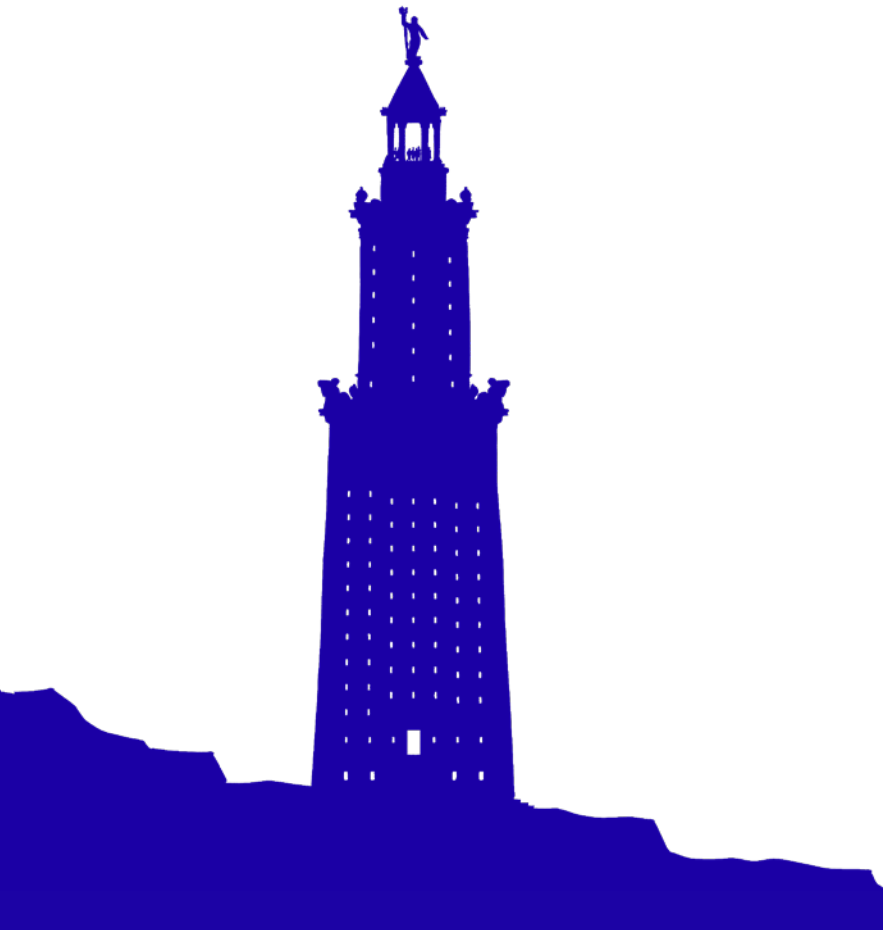
Anna Sophie holds an MA in International Studies from Seoul National University and a BSc in International Business from the University of Groningen.

James Shires

James Shires is the Co-Director of Virtual Routes. He is also a Fellow with The Hague Program on International Cyber Security. He was previously a Senior Research Fellow in Cyber Policy at Chatham House, and before that an Assistant Professor in Cybersecurity Governance at the Institute of Security and Global Affairs, University of Leiden.

He has written widely on issues of cybersecurity and international politics, including cybersecurity expertise, digital authoritarianism, spyware regulation, and hack-and-leak operations. He is the author of The Politics of Cybersecurity in the Middle East (Hurst/Oxford University Press, 2021), and co-editor of Cyberspace and Instability (Edinburgh University Press, 2023). A full list of publications is available at jamesshires.com

Table of Contents



Executive Summary

Artificial intelligence (AI) is increasingly embedded in critical systems across government, industry, and society. Most AI systems depend on a complex ecosystem of datasets, models, software libraries, and infrastructure sourced from multiple external actors. Much of this ecosystem is built around machine learning (ML), where systems are trained on data and reused across applications. This report describes these interconnected dependencies as the ML and AI supply chain.

This ecosystem enables rapid innovation by allowing developers to reuse existing components. But it also introduces new and often underappreciated security risks. Vulnerabilities in widely used upstream components can propagate across many downstream systems, creating systemic exposure that is difficult to detect and contain.

This report examines how security risks emerge within the machine learning and AI supply chain and what can be done to mitigate them. It makes three central arguments.

First, many of the most significant AI security risks arise before deployment.
While public attention often focuses on how AI systems behave once deployed, vulnerabilities are frequently introduced earlier – during data collection, model training, and the integration of third-party components. Weaknesses introduced at these stages can create security exposures that are difficult to detect, attribute, or remediate later.

Second, scale and reuse amplify systemic risk.
AI development depends heavily on shared datasets, pre-trained models, and open-source tools. These components are often trusted implicitly and reused across many systems. When widely adopted resources are compromised – whether through data manipulation, malicious code, or hidden backdoors – the effects can cascade across multiple organisations and sectors.

Third, effective mitigation depends on managing – not eliminating – interdependence.
Reuse and openness are essential to AI innovation and cannot be removed without significant cost. Instead, securing the AI supply chain depends on three core principles: visibility, control, and accountability. Visibility helps organisations understand dependencies and identify where vulnerabilities may be introduced. Control – through access management, verification, and secure configuration – reduces the likelihood that compromised components propagate unnoticed. Accountability, through testing, monitoring, and documentation, improves detection, response, and resilience when failures occur.

To support this analysis, the report breaks the machine learning and AI supply chain into three levels:

- **The data level**, where AI systems rely on large datasets as the foundation for training and performance, creating vulnerabilities when data is manipulated, poisoned, corrupted, or insufficiently verified.
- **The model level**, where training processes, pre-trained models, and code dependencies introduce risks related to hidden functionality and unsafe execution.
- **The deployment level**, where systems interact with real-world inputs and users, creating opportunities for manipulation and information extraction.

Across these levels, malicious actors exploit trust, complexity, and limited visibility. They may involve poisoning training data, substituting malicious components, embedding hidden behaviours in models, or manipulating system inputs after deployment.

For policymakers, these dynamics have important implications. AI security cannot be addressed solely through regulation focused on model behaviour and outputs, or through post-deployment oversight alone. It also requires attention to upstream dependencies, development practices, and the broader ecosystem in which AI systems are built and maintained. This includes issues such as data governance, software supply-chain security, standards for documentation and provenance, the resilience of shared infrastructure, and mechanisms that improve visibility, verification, and accountability across different stages of AI development and deployment.

The report concludes that securing AI systems requires a shift in perspective – from viewing them as standalone technical artefacts to understanding them as part of a distributed and evolving supply chain. Managing risk in this environment will require coordinated action across technical, organisational, and policy domains. The objective is not to eliminate risk entirely, but to make risks more visible, constrain their impact, and strengthen resilience throughout the broader ecosystem.

> Securing the AI supply chain depends on three core principles: visibility, control, and accountability.

Introduction

In March 2026, a cybersecurity incident affecting widely used AI development tools exposed a structural vulnerability in how modern AI is built. Rather than targeting a single system, attackers compromised a trusted upstream component, allowing the attack to propagate through downstream applications via routine processes – software updates, automated pipelines, and shared dependencies.

At the centre of the incident was LiteLLM, an open-source library that connects applications to multiple AI services through a single interface, rather than building a separate integration for each AI provider. Because it mediates between applications and external models, it often handles sensitive credentials such as API keys. By early 2026, it had become a critical piece of infrastructure for organisations building AI-powered applications.

Attackers exploited a weakness in a tool used within LiteLLM's development pipeline, gained access to credentials, and modified its code and distribution. When LiteLLM later relied on this dependency, it unknowingly incorporated malicious code, enabling attackers to publish a tampered version using legitimate credentials.

The significance lies not in a single vulnerability, but in what it reveals about how AI systems are constructed. Modern AI development depends on extensive reuse: of datasets, pre-trained models, software libraries, and cloud-based infrastructure. These components are often integrated through automated workflows and maintained by different actors across organisational and geographic boundaries. As a result, trust is distributed – and often implicit – across a wide network of dependencies.

This report examines this system as a machine learning and AI *supply chain*: the interconnected set of dependencies through which modern AI systems are developed, trained, deployed, and maintained. Viewing AI through this lens shifts the focus from individual models to the broader ecosystem in which they are developed and maintained. It highlights how vulnerabilities can be introduced upstream, remain undetected during development, and only become visible once systems are deployed or reused.

For policymakers, this creates a core challenge. AI systems are increasingly integrated into critical functions, yet the dependencies that underpin them are often opaque and outside the control of those who deploy them. This limits visibility, complicates accountability, and weakens existing approaches to oversight.

The sections that follow examine how vulnerabilities emerge across the machine learning and AI supply chain, how attackers can exploit them at different stages of the supply chain, and what practical measures can strengthen security and resilience.

The Machine Learning and AI Supply Chain

To understand where risks emerge, it is necessary to examine how AI systems are built. This report uses the term 'AI supply chain' to describe that process: the interconnected set of activities and dependencies involved in developing, training, and deploying AI systems.[1]

While artificial intelligence encompasses a wide range of approaches, the risks examined here are primarily associated with machine learning, which underpins most modern AI systems. Unlike traditional software, machine learning systems are not defined solely by code. Their behaviour is shaped by data and training processes, meaning that trust depends not only on software integrity, but also on data provenance and how models are trained.

From a supply-chain perspective, this expands the scope of dependencies beyond code to include data and models, where visibility and verification are more limited. Machine learning systems rely on a broad ecosystem of external resources: datasets, software libraries, pre-trained models, cloud infrastructure, and specialised tooling used to manage data and training workflows. These components are often developed and maintained by different actors, integrated through automated processes, and reused across multiple projects. Dependencies are therefore layered and distributed, creating both efficiency gains and new points of vulnerability.

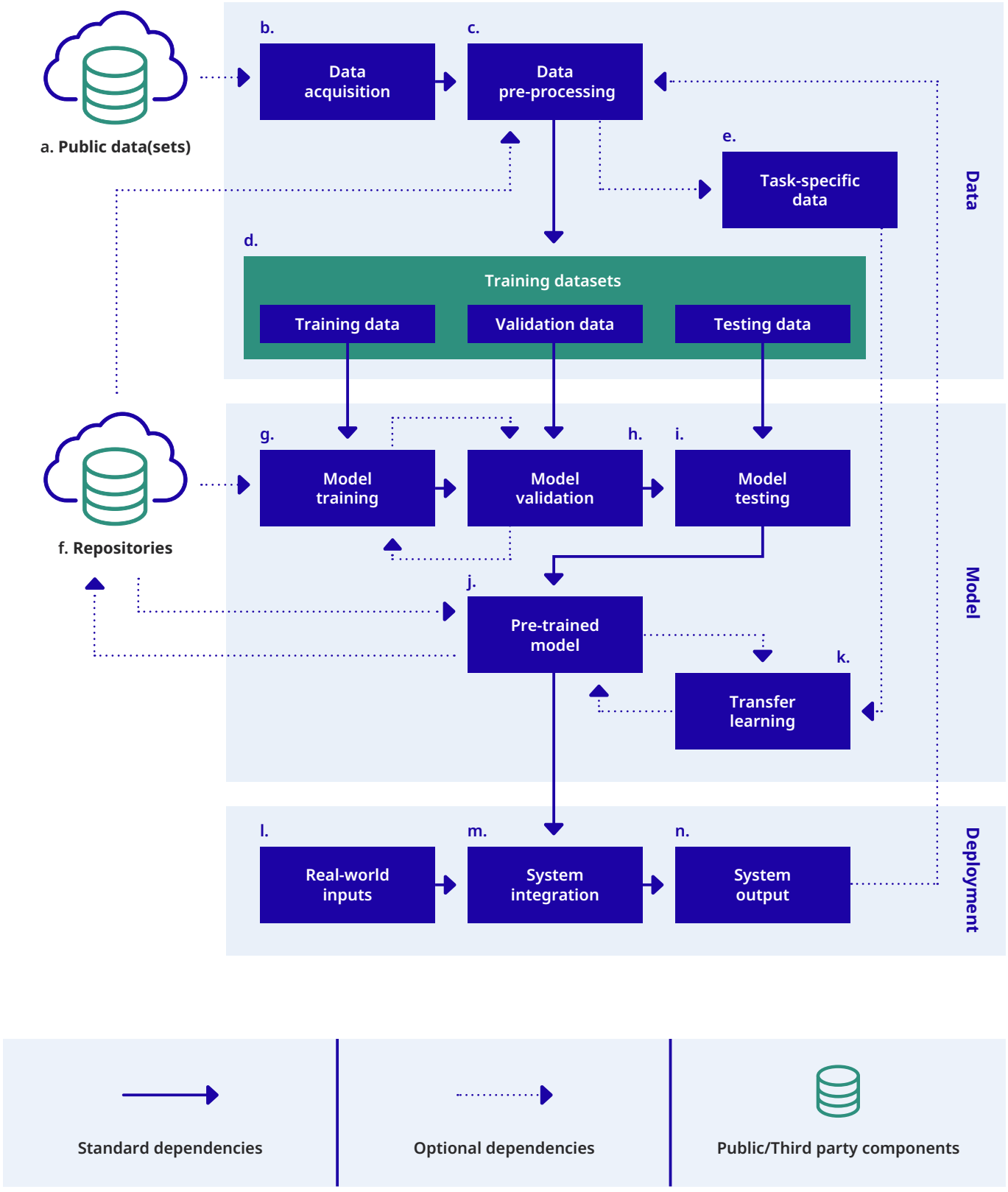
The machine learning and AI supply chain can be understood at three interrelated stages:

- **Data level**: data collection, sourcing, labeling, preprocessing, storage, and reuse.
- **Model level**: model design, training, validation, testing, and reuse of pre-trained models.
- **Deployment level**: integration into applications and interaction with real-world inputs and users.

In practice, AI development is not linear. Models may be built on existing components, updated continuously, or extended through live data inputs. This framework provides a baseline for understanding where dependencies form and where risks emerge.

[1] AI is generally understood as non-human intelligence characterised by its capacity to replicate human cognitive abilities, such as pattern recognition, natural language understanding, and adaptive learning from experience. It is not one technology, AI is rather a collection of distinct fields that develop different technologies which enable the emergence of intelligent behaviour. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, 4th edn (Pearson, 2021).

Figure 1: Key Components and Stages in the Machine Learning and AI Supply Chain



The Data Level

Data forms the foundation of the machine learning and AI supply chain and strongly influences how AI systems perform. Training data is often collected from publicly available web content, licensed from external providers such as news or image platforms, or taken from existing third-party datasets (part a in Figure 1).[2] Many of these datasets are designed for specific tasks. For example, ImageNet is used to train systems to recognise objects in images, the Common Objects in Context (COCO) dataset helps systems identify multiple objects within a scene, and LibriSpeech supports speech recognition development. These shared datasets reduce the cost of training AI systems and allow developers to compare performance across models. However, because they are widely reused, they also become important upstream dependencies: vulnerabilities or manipulation introduced into a commonly used dataset can affect many downstream systems.

Spotlight case study: ImageNet

Before 2009, progress in computer vision was constrained by the lack of large, well-labeled datasets. Developed under the leadership of Fei-Fei Li at Stanford University, ImageNet provided a large, standardised dataset that enabled rapid advances in visual recognition. Today, it contains over 14 million labeled images organised into thousands of categories.[3]

ImageNet's categories were derived from WordNet, a linguistic database that groups words by meaning. As a result, the dataset reflects the assumptions embedded in its taxonomy and labeling process.

ImageNet also became a central benchmark for the field through the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), which ran from 2010 to 2017. The 2012 success of AlexNet marked a turning point, demonstrating the effectiveness of deep learning and accelerating its adoption across AI research and industry.[4]

From a supply-chain perspective, ImageNet illustrates both the benefits and risks of widely reused upstream components. Its availability enabled rapid innovation and standardisation, but its widespread use also means that its assumptions, biases, and potential flaws propagate across many downstream systems. Once embedded in widely used models and benchmarks, these characteristics can persist even as systems are adapted and reused in new contexts.

---

[2] Sergi Gálvez Duran, *Mapping Relevant Data Collection Mechanisms for AI Training*, OECD Artificial Intelligence Papers 48 (OECD, 2025), https://www.oecd.org/content/dam/oecd/en/publications/reports/2025/10/mapping-relevant-data-collection-mechanisms-for-ai-training_62921889/3264cd4c-en.pdf.

[3] Jia Deng et al., 'ImageNet: A Large-Scale Hierarchical Image Database', *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, 248–55, https://doi.org/10.1109/CVPR.2009.5206848.

[4] Olga Russakovsky et al., 'ImageNet Large Scale Visual Recognition Challenge', *International Journal of Computer Vision* 115, no. 3 (2015): 211–52, https://doi.org/10.1007/s11263-015-0816-y.

Beyond public datasets, organisations increasingly rely on proprietary data and large-scale collection from online platforms (part b in Figure 1).[5] Much of this data was not originally created for machine learning purposes, raising legal, ethical, and governance questions about consent, ownership, and appropriate use. Disputes involving platforms such as Reddit, YouTube, and LibGen highlight how data sourcing practices can affect trust, transparency, and accountability within the broader AI ecosystem.[6]

From a supply-chain perspective, these practices also introduce security risks. When data pipelines are opaque or contested, it becomes difficult to assess the origin, handling, and integrity of the data, increasing the likelihood that manipulation or errors go undetected.

Before training, data usually undergoes preprocessing (part c in Figure 1), where raw inputs are cleaned, organised, filtered, and converted into formats that models can use. In language models, this often includes tokenisation: breaking text into smaller units ('tokens') that systems process during training and inference. Although preprocessing is often treated as a technical preparation step, it can significantly influence model performance and creates another point in the pipeline where data may be altered – intentionally or unintentionally – without being visible later.

Training data is usually divided into three sets (part d in Figure 1). The training set is the data the model learns from directly. The validation set is used during development to monitor and improve performance. The test set is kept separate until the end of training to evaluate how well the model performs on new, unseen data. These divisions in training data are central to AI development, but they also create multiple points where data is stored, transferred, modified, and reused. At each stage, questions of data integrity, provenance, and access control become important.

The Model Level

At the model level, AI systems are developed by training computational models to learn patterns from data and produce outputs such as classifications, predictions, or generated content.[7] This stage of the AI supply chain includes model design, training, and evaluation.

---

[5] For a more in depth discussion on the human resources required for labeling and other relevant tasks: Kate Crawford, *Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence* (New Haven, 2021).

[6] Reddit's decision to license user-generated posts and comments to AI companies, including a $60 million deal with Google, has drawn criticism from users and regulators alike. The US Federal Trade Commission (FTC) initiated an investigation into Reddit's data licensing practices, questioning the platform's transparency and user consent mechanisms. In a separate legal action, Reddit sued Anthropic, alleging that the AI company scraped its content without permission to train its chatbot, Claude. Similarly, YouTube's vast repository of videos has been utilised by tech giants like Google, Apple, and Nvidia to train AI models. Many content creators were unaware that their videos were being used in this manner, leading to concerns about intellectual property rights and the lack of compensation. On the audio front, Amazon's Alexa has faced legal challenges over its data collection practices. The company settled with the FTC for $25 million after allegations that it violated children's privacy laws by retaining voice recordings without proper consent. Ty Roush, 'Reddit Accuses Anthropic Of Taking User Data Without Consent', Forbes, 4 June 2025, https://www.forbes.com/sites/tylerroush/2025/06/04/reddit-claims-anthropic-trained-ai-models-using-its-users-data-without-consent/; Annie Gilbertson and Alex Reisner, 'Apple, Nvidia, Anthropic Used Thousands of Swiped YouTube Videos to Train AI', Tags, Wired, 16 July 2024, https://www.wired.com/story/youtube-training-data-apple-nvidia-anthropic/; Ella Creamer, '"Meta has stolen books': authors to protest in London against AI trained using 'shadow library"', *The Guardian*, 3 April 2025, https://www.theguardian.com/books/2025/apr/03/meta-has-stolen-books-authors-to-protest-in-london-against-ai-trained-using-shadow-library.

[7] The developing of an AI system generally follows one of three pathways; (1) train a model from scratch, (2) download a pre-trained model from an open-source model hub and use it directly, or (3) download a pre-trained model from an open-source model hub and fine-tune it for a specific task. Qiang Hu et al., 'Large Language Model Supply Chain: Open Problems From the Security Perspective', arXiv:2411.01604, preprint, arXiv, 3 November 2024, https://doi.org/10.48550/arXiv.2411.01604.

A machine learning model combines executable code with numerical parameters that determine how inputs are transformed into outputs. These parameters – often referred to as weights or coefficients – are adjusted during training based on statistical patterns in the training data, allowing the model to estimate the probability that certain inputs correspond to particular outputs. Depending on the type of system, this can range from relatively simple decision rules to highly complex representations across large numbers of variables.

During training (part g in Figure 1), parameters are iteratively updated using training data so that predictions align with expected outcomes. This process depends not only on the data, but also on training code, configuration choices, and the computational environment. From a supply-chain perspective, risks are therefore not confined to the model itself, but extend to the broader environment in which training occurs.

Following training, models undergo validation (part h in Figure 1) to assess whether they generalise beyond the data they were trained on. Validation helps guide configuration decisions, but practices vary widely across organisations, meaning weaknesses introduced during training may not always be identified.

Testing provides a final assessment of performance on previously unseen data (part i in Figure 1). While intended to approximate real-world conditions, testing captures only a subset of possible scenarios and depends on how well test data reflects actual use.

In practice, model development does not occur in isolation. Training and evaluation take place within a broader ecosystem of third-party libraries, frameworks, and pre-trained models (part f in Figure 1).[8] Widely used tools such as TensorFlow, PyTorch, and Scikit-learn provide standardised building blocks, but also embed assumptions about model design and configuration.

> “Trust in the final system depends on the integrity of a broader network of components.”

These dependencies are often layered within the model level: a single training pipeline may incorporate multiple external components, each maintained and updated independently. Model development is therefore a cumulative process rather than a self-contained one. Trust in the final system depends on the integrity of a broader network of components, many of which may be outside the direct control or visibility of the organisation deploying it.

[8] Model code is typically maintained in version-controlled repositories such as GitHub, GitLab, or Bitbucket. These repositories store training scripts, model architectures, data preprocessing pipelines, configuration files, and evaluation metrics, and provide a record of changes over time. As such, they form the organisational backbone for subsequent stages of model training, validation, and deployment.

A central mechanism enabling this reuse is transfer learning (part k in Figure 1). Instead of training models from scratch, developers adapt pre-trained models to new tasks. This reduces time and resource requirements, but also deepens reliance on upstream components, extending the chain of dependencies that shape model behaviour.

The Deployment Level

At the deployment level, trained models are integrated into systems that operate in real-world environments. Rather than functioning in isolation, models are embedded within applications that combine AI inference with user interfaces, data pipelines, operational rules, automated processes, and supporting infrastructure. These surrounding systems determine how inputs are handled, how outputs are interpreted, and how model decisions influence real-world actions.

Once deployed, systems receive inputs from dynamic environments that are more difficult to control than those used during training and testing (part l in Figure 1). These inputs may include user-generated content, sensor data, uploaded files, camera feeds, or continuous data streams. In many systems, inputs are also filtered, transformed, enriched with external information, or routed through retrieval systems before reaching the model. Some deployment environments additionally use user interactions and feedback to refine or retrain models over time.[9]

Model outputs are then incorporated into broader application workflows (part n in Figure 1), where they may inform decisions, trigger automated actions, support human judgment, or shape recommendations presented to users. Outputs are often stored, logged, shared with other systems, or used in subsequent processing steps. As a result, the effects of AI systems depend not only on the model itself, but also on how outputs are interpreted, integrated, and acted upon in operational settings.

From a supply-chain perspective, deployment shifts the focus from how systems are built to how they operate in real-world conditions. Models function in dynamic environments where unexpected, manipulated, or adversarial inputs are unavoidable and where many supporting systems remain outside the direct control of developers. Embedded within broader technical and organisational environments – including APIs, cloud services, data pipelines, external vendors, operational processes, and human oversight – AI systems are exposed to additional vulnerabilities and dependencies. With real-world use, weaknesses introduced earlier in the supply chain often become visible and new risks emerge.

[9] Together, these processes extend the AI supply chain beyond the model itself, introducing dependencies on external data sources, APIs, platforms, sensors, and supporting infrastructure.

Adversarial Attacks Against the Supply Chain

With the AI supply chain framework in place, this section examines how adversaries exploit weaknesses across different stages of AI development and deployment. Rather than targeting isolated systems, these attacks take advantage of the same characteristics that underpin modern AI development: extensive reuse of shared components, reliance on upstream dependencies, and highly automated development and deployment processes. This means compromises introduced at one stage can spread across multiple downstream systems, often remaining undetected until models are deployed or widely reused.

Attacks emerge at different points in the supply chain (see Figure 2). At the data level, adversaries manipulate the information used to train models. At the model level, they target training processes, model artefacts, or software dependencies. At the deployment level, they exploit how systems interact with real-world inputs, users, and connected systems.[10]

> “Compromises introduced at one stage can spread across multiple downstream systems, often remaining undetected until models are deployed or widely reused.”

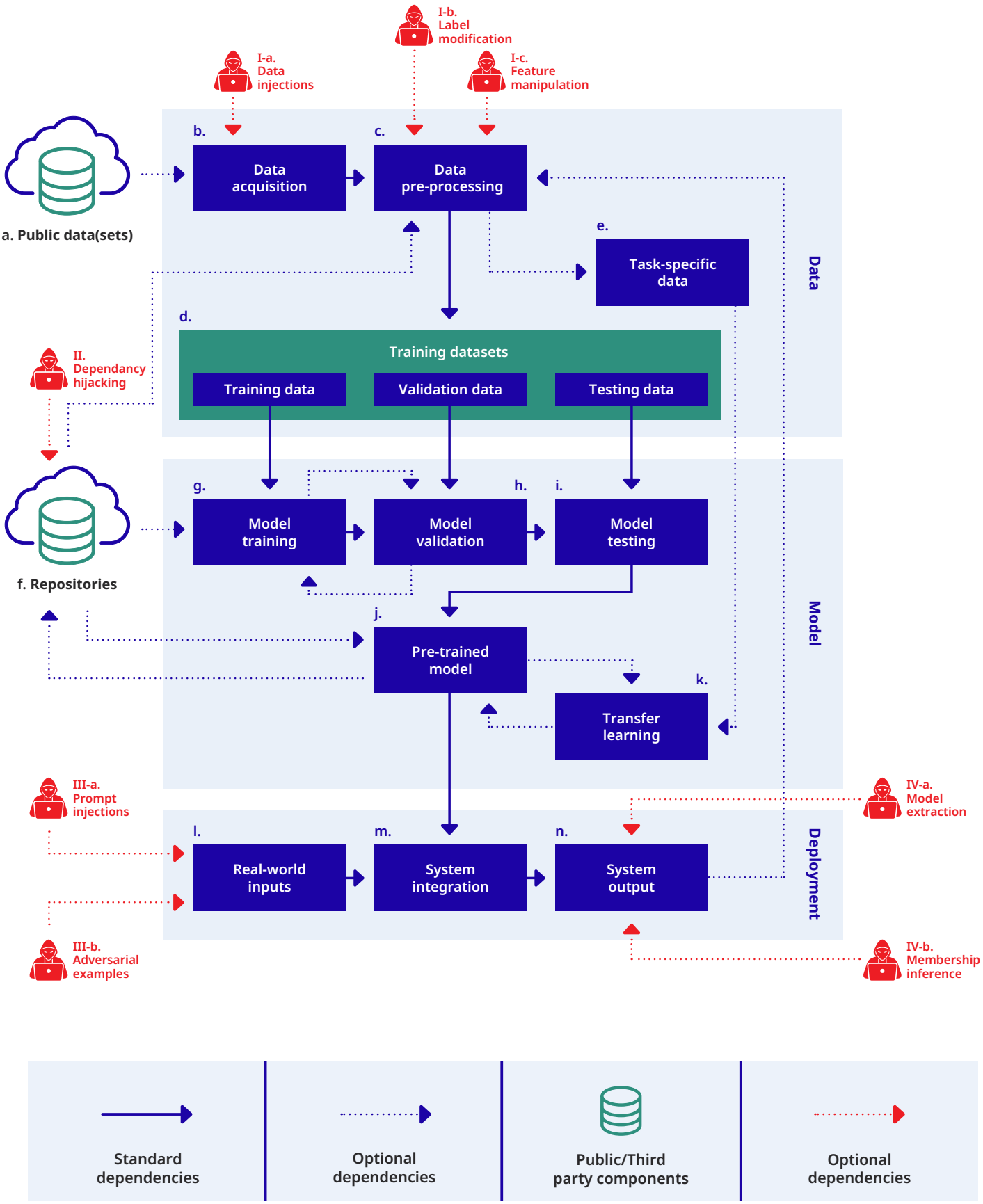
Adversarial Attacks at the Data Level

At the data level, the most significant risks relate to data integrity. These attacks – commonly referred to as ***data poisoning*** – do not directly alter model code or parameters. Instead, they manipulate the data from which models learn, indirectly shaping how systems behave.

The effects of poisoning can be difficult to detect. Machine learning models often operate as complex statistical systems, in which the relationship between training data and model behaviour is not easily interpretable. Manipulated data may therefore influence model outputs without leaving obvious traces in the final system. In some cases, poisoned behaviour may only appear under specific conditions or trigger inputs, allowing compromised models to appear normal during routine testing while behaving differently in deployment.

[10] These categories are analytically useful, but – as previously noted – the stages are closely connected. For example, manipulated training data or compromised model components introduced upstream may only produce visible effects once systems are deployed and interacting with users. Understanding AI supply-chain security therefore requires examining not only individual vulnerabilities, but also how weaknesses introduced at one stage can spread across downstream systems and environments.

Figure 2: Types of Adversarial Attacks Across the Machine Learning and AI Supply Chain



Because training data is frequently reused across projects, the effects of poisoning can extend beyond a single model. Compromised data may influence multiple downstream systems, particularly where models are fine-tuned or retrained on shared datasets.

A well-known illustration is Microsoft's chatbot Tay, launched in 2016. Designed to learn from real-time user interactions, Tay incorporated external inputs directly into its training process. Coordinated users supplied harmful content, which the system absorbed and reproduced. While unusually visible, the case demonstrates a broader point: when data pipelines rely on untrusted inputs, attackers can influence system behaviour without accessing the underlying model.[11]

In practice, data poisoning takes several forms.

First, attackers may ***inject new data***, such as fabricated or strategically crafted samples, into training pipelines. This is particularly effective when targeting widely reused datasets or open submission channels, where even small amounts of manipulated data can have disproportionate effects (part i-a in Figure 2).[12]

A group of researchers demonstrated that poisoning web-scale datasets can be both practical and inexpensive when attackers exploit weaknesses in data collection and curation processes.[13] Studying datasets such as LAION, a large-scale image-text dataset widely used to train generative AI models , and COYO, a similarly large collection of image-caption pairs gathered from the public web, they showed how adversaries could introduce malicious samples by manipulating upstream web content that is later incorporated into training datasets. Their findings highlight a broader supply-chain challenge: compromising a relatively small number of upstream data sources can affect datasets that are subsequently reused across many downstream systems.

Second, attackers may ***modify existing data***, often through label manipulation. In these cases, otherwise valid data is assigned incorrect labels – for example, misclassifying images or malware samples. This can degrade overall performance or produce targeted errors. In 2020, for instance, researchers observed adversaries uploading near-duplicate ransomware samples that were automatically mislabeled, reducing the effectiveness of security systems trained on that data (part i-b in Figure 2).[14]

Third, attackers may engage in ***feature manipulation*** by influencing how models learn without altering labels (part i-c in Figure 2). In ***clean-label poisoning***, inputs appear legitimate but are designed to subtly shape internal model representations. Because these samples pass standard

[11] Helena Horton, 'Microsoft Deletes "Teen Girl" AI After It Became a Hitler-Loving Sex Robot within 24 Hours', *Telegraph*, 24 March 2016, https://www.telegraph.co.uk/technology/2016/03/24/microsofts-teen-girl-ai-turns-into-a-hitler-loving-sex-robot-wit/; Peter Lee, 'Learning from Tay's Introduction', *Official Microsoft Blog*, 25 March 2016, https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/.

[12] Alexandra Souly et al., 'Poisoning Attacks on LLMs Require a Near-Constant Number of Poison Samples', arXiv:2510.07192, preprint, arXiv, 8 October 2025, https://doi.org/10.48550/arXiv.2510.07192.

[13] Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr, 'Poisoning Web-Scale Training Datasets Is Practical', 2024 IEEE Symposium on Security and Privacy (SP), May 2024, https://doi.org/10.1109/SP54263.2024.00179

[14] 'VirusTotal Poisoning', MITRE ATLASTM, accessed 14 May 2026, https://atlas.mitre.org/studies/AML.CS0002.

validation checks, their effects may only become visible under specific conditions after deployment. By contrast, more overt 'dirty-label' attacks are easier to detect but can still disrupt performance.

Across these approaches, the common feature is the exploitation of trust in data pipelines. Training data is often aggregated from multiple sources, processed automatically, and reused across systems. This makes it difficult to verify integrity at scale. As a result, even limited manipulation at the data level can propagate widely and persist over time, shaping model behaviour long after the initial point of compromise.

Adversarial Attacks at the Model Level

At the model level, adversarial attacks target the components and processes through which models are built, shared, and loaded. Rather than altering application code directly, attackers exploit the ecosystem of packages, pre-trained models, and configuration tools that underpin modern AI development. This makes the model level a particularly attractive target for supply-chain attacks.

A key entry point is ***dependency hijacking*** – the substitution of malicious artefacts for legitimate ones (part ii in Figure 2). This often occurs through 'typosquatting', where attackers publish packages or models with names similar to widely used components, increasing the likelihood of accidental installation. A newer variant, sometimes referred to as 'slopsquatting', exploits errors introduced by AI coding assistants. When these systems generate plausible but non-existent package names, attackers can register them and distribute malicious code that propagates as AI-generated scripts are reused.

> Because training data is frequently reused across projects, the effects of poisoning can extend beyond a single model.

Once introduced, malicious artefacts typically rely on how they are loaded and executed. Configuration files, which specify how models are retrieved and run, are often treated as passive metadata. In practice, some formats – such as YAML – support advanced features that can trigger unintended behaviour when parsed unsafely, including the execution of attacker-controlled code.

A common mechanism through which dependency hijacking is executed is through ***model serialisation***. Serialised model files are designed to enable portability and reuse, but some formats execute code when they are loaded. This means that simply loading a model can trigger malicious instructions. While this risk is often discussed in the context of AI security, it reflects a broader software security problem: executable content embedded within trusted artefacts. In practice, unsafe loading mechanisms and insecure handling of serialised objects have long created vulnerabilities in conventional software systems.

The machine learning and AI supply chain, however, amplifies these risks because models and associated artefacts are frequently downloaded from public repositories, reused across projects, and integrated through highly automated workflows. Researchers have identified compromised models distributed through public repositories that, when loaded, opened reverse shells and granted remote access to attackers.

Not all model-level attacks rely on code execution. Some instead exploit the widespread sharing and reuse of pre-trained models within the supply chain. In these cases, adversaries embed hidden backdoors during model training that persist as models are shared, adapted, and fine-tuned for downstream applications. The compromised model behaves normally under standard evaluation, but produces manipulated outputs when specific trigger conditions are present. Because organisations using pre-trained models often have limited visibility into how those models were developed, modified, or trained, such backdoors can be difficult to detect and may propagate across multiple downstream systems before discovery.

Across these approaches, the common feature is the exploitation of implicit trust in model artefacts and the mechanisms used to integrate them. Because these compromises occur upstream, a single malicious dependency or model can propagate across multiple systems before detection, broadly affecting integrity, confidentiality, and availability.

Adversarial Attacks at the Deployment Level

At the deployment level, adversarial attacks exploit how AI systems interact with their environment through application interfaces. Once a model is deployed, attackers no longer need access to training data or internal components. Instead, they operate through inputs and outputs, manipulating how systems interpret information or extracting sensitive data from their behaviour.

Two broad categories of attacks dominate at this stage: ***input manipulation*** and ***output-based extraction***.

The first involves crafting inputs that cause the system to behave incorrectly while appearing to function normally. In language-model applications, this includes ***prompt injection***, where attackers embed hidden instructions into inputs (part iii-a of Figure 2). These may be added directly to user prompts or indirectly through external content such as emails, documents, or web pages. Because many systems combine user queries with retrieved context, attackers can exploit this process to override system instructions or bypass safeguards.

The EchoLeak attack against Microsoft 365 Copilot illustrates this risk.[15] Researchers showed that a malicious prompt embedded in a routine business email could be ingested by the system and trigger the exfiltration of sensitive internal data. The attack did not compromise the model itself; it

[15] Trend Micro, "Preventing Zero-Click AI Threats: Insights from EchoLeak," Accessed 15 July 2026, https://www.trendmicro.com/en_us/research/25/g/preventing-zero-click-ai-threats-insights-from-echoleak.html

succeeded by manipulating how inputs were assembled and interpreted during deployment.

More broadly, input manipulation includes adversarial examples, which affect systems across modalities. Small, often imperceptible changes to inputs – whether images, audio, or text – can cause models to produce incorrect outputs. These attacks exploit the statistical nature of machine learning, allowing adversaries to bypass safeguards without triggering obvious failures.

The second category focuses on extracting information from model outputs. Even when models are deployed as black boxes, their behaviour can reveal sensitive information. In ***model extraction*** attacks, adversaries systematically query a system and use its responses to reconstruct an approximate version of the underlying model (part iv-a of Figure 2). This can undermine intellectual property protections and enable unauthorised replication of proprietary systems.

Such attacks have been demonstrated in practice, with researchers showing that models accessible via public APIs can be replicated through large volumes of structured queries. Individual interactions may appear benign, but harmful activity emerges over time through repeated probing.[16]

Related techniques target the privacy of training data. ***Membership inference attacks*** aim to determine whether a specific data point was included in a model's training set, while ***model inversion attacks*** attempt to reconstruct sensitive attributes or approximate training examples from outputs (part iv-b of Figure 2).

Beyond these two broad categories of input manipulation and output-based extraction, vulnerabilities may also arise from the infrastructure used to deploy and operate AI systems. In 2025, researchers at SentinelOne and Censys identified thousands of publicly exposed deployments of Ollama, a widely used platform for running large language models on local computers and servers[17]. Many of these systems had been configured to accept remote connections and were linked to external tools or local resources. In such cases, attackers did not need to manipulate model behaviour or extract model information. Instead, access to the deployment infrastructure could allow them to invoke connected tools, interact with local systems, or consume computational resources remotely.

Taken together, deployment-level attacks highlight a shift in how AI systems are targeted. Rather than compromising internal components, adversaries exploit the observable behaviour of systems in operation. This makes detection more challenging, as harmful activity may arise not from single interactions, but from patterns of use over time.

[16] Florian Tramèr et al., 'Stealing Machine Learning Models via Prediction APIs', *arXiv.Org*, 9 September 2016, https://arxiv.org/abs/1609.02943v2.

[17] Gabriel Bernadett-Shapiro and Silas Cutler, 'Silent Brothers: Ollama Hosts Form Anonymous AI Network Beyond Platform Guardrails', SentinelOne Labs, 2025, https://www.sentinelone.com/labs/silent-brothers-ollama-hosts-form-anonymous-ai-network-beyond-platform-guardrails/

Towards Mitigation

The vulnerabilities described in the previous section highlight a central challenge: AI systems cannot be secured at a single point. Risks arise across the data, model, and deployment levels, often interacting in ways that make them difficult to detect and contain.

The same characteristics that enable AI to scale – reuse, automation, external dependencies, and shared infrastructure – also shape how mitigation must be designed. These features cannot be removed without undermining innovation; they must be managed more deliberately.

In practice, this requires moving away from implicit trust in upstream components towards greater visibility, control, and accountability throughout the machine learning and AI supply chain. It also means recognising that measures at any single stage are necessarily partial. Weaknesses introduced upstream may only become visible at deployment, while controls applied at deployment cannot address risks already embedded in data or models.

> Mitigation must therefore be approached as a layered process spanning the different stages of AI development and deployment.

At the data level, this involves managing how data is sourced, validated, and maintained. At the model level, it requires securing how models and dependencies are introduced and executed. At the deployment level, it focuses on managing system behaviour in open environments and improving the ability to monitor and constrain it in practice.

The sections that follow examine mitigation measures at each level in turn. The aim is not to eliminate risk entirely, but to make it more visible, limit its impact, and strengthen the capacity to detect and respond to failures.

Mitigation at the Data Level

At the data level, mitigation focuses on two priorities: preventing untrusted data from entering training pipelines, and ensuring that data already in use cannot be altered without detection. This directly addresses the risks of data injection, label manipulation, and more subtle forms of poisoning.

A key point of exposure is ***data ingestion***, particularly where datasets are sourced from external or continuously updated environments. Pipelines that rely on web scraping, open contributions, or

aggregated third-party inputs are especially vulnerable. In these settings, attackers can introduce manipulated data that blends into legitimate sources. Reducing this risk requires greater scrutiny of ***data provenance*** – not only where data originates, but how it is collected, combined, and maintained.[18]

In practice, this can be supported through structured documentation. Approaches such as Google's 'data cards' and dataset documentation standards used on platforms like Hugging Face provide information on data sources, collection methods, and intended use, improving transparency and traceability.[19] For policymakers, these practices highlight the importance of standards for documentation and auditability throughout the AI supply chain.

Once data is ingested, mitigation shifts to identifying irregularities within datasets. Techniques such as outlier detection or clustering can help flag anomalous data, but they are most effective against broad or noisy manipulation[20]. More targeted attacks – designed to resemble legitimate data – may evade these checks, limiting the effectiveness of purely technical safeguards.[21]

Labeling processes are another critical vulnerability. Because labels are often assigned through a mix of automated systems and human annotation, they can introduce both errors and opportunities for manipulation. Strengthening label integrity requires adding verification and redundancy, such as cross-checking annotations, auditing high-risk categories, and testing for inconsistencies across datasets. These measures are most effective when focused on areas where errors would have the greatest downstream impact.

More subtle threats, such as clean-label poisoning, are harder to detect because individual data points appear legitimate. In these cases, mitigation shifts from inspecting data to evaluating its effects on model behaviour – for example, by testing models under adversarial conditions or comparing outputs across independently sourced datasets.

Across these measures, a common requirement is traceability. Organisations need to be able to track how data has been collected, modified, and reused over time. This includes maintaining versioned datasets, recording provenance, and controlling who can modify data pipelines. Without this visibility, it becomes difficult to distinguish between normal variation and deliberate interference.

Additional safeguards aim to reduce the exposure of sensitive information during training. Techniques such as data masking, aggregation, pseudonymisation, and differential privacy can

---

[18] Justin Sherman, *Securing Data in the AI Supply Chain* (Atlantic Council, 2025), https://www.atlanticcouncil.org/in-depth-research-reports/issue-brief/securing-data-in-the-ai-supply-chain/.

[19] Mahima Pushkarna et al., 'Data Cards: Purposeful and Transparent Dataset Documentation for Responsible AI', arXiv:2204.01075, preprint, arXiv, 3 April 2022, https://doi.org/10.48550/arXiv.2204.01075.

[20] Sridhar Venkatesan et al., 'Poisoning Attacks and Data Sanitization Mitigations for Machine Learning Models in Network Intrusion Detection Systems', MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM), November 2021, 874–79, https://doi.org/10.1109/MILCOM52596.2021.9652916; Jacob Steinhardt et al., 'Certified Defenses for Data Poisoning Attacks', Advances in Neural Information Processing Systems 30 (2017), https://proceedings.neurips.cc/paper/2017/hash/9d7311ba459f9e45ed746755a32dcd11-Abstract.html.

[21] Apostol Vassilev et al., *Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations*, NIST Artificial Intelligence (AI) 100-2 E2025 (National Institute of Standards and Technology, 2025), https://doi.org/10.6028/NIST.AI.100-2e2025; Souly et al., 'Poisoning Attacks on LLMs Require a Near-Constant Number of Poison Samples'.

reduce the likelihood that models memorise individual data points or retain directly identifiable information.[22] This lowers the risk of extraction attacks or unintended disclosure later in deployment. These approaches are already used in production by organisations such as Apple and Google, though they often involve trade-offs in accuracy, utility, and system performance.[23]

Protecting data integrity also depends on securing how data is stored and transmitted. Techniques such as hashing and digital signatures can detect unauthorised changes, while verifying dependencies reduces the risk of introducing compromised inputs. These practices align with guidance from organisations such as the US National Institute of Standards and Technology (NIST), which emphasises integrity checks across software and data pipelines.

Finally, the environments in which data pipelines operate must be secured. This includes restricting access to sensitive data, using short-lived credentials, and limiting unnecessary external connections. These controls reduce the risk of data exfiltration or tampering and are reflected in broader governance frameworks, including the European Union's General Data Protection Regulation (GDPR).[24]

Mitigation at the Model Level

At the model level, risks arise not from how models are used, but from how model artefacts and dependencies are introduced and handled within development pipelines. As discussed earlier, these risks stem from three main issues: the substitution of dependencies or pre-trained models, unsafe execution during loading, and hidden backdoors within otherwise legitimate systems. Mitigation therefore focuses on reducing implicit trust in external components, securing how they are executed, and strengthening validation before deployment.

A primary area of exposure is how models and software packages are sourced. Reducing this risk requires moving from implicit to explicit trust – for example, by pinning dependencies to specific versions and verifying their integrity using cryptographic hashes.[25] More structured approaches, such as Google's Supply-chain Levels for Software Artifacts (SLSA) framework, provide mechanisms to verify provenance and integrity across build and deployment processes.[26]

Organisations should also assess the reliability of external sources, prioritising repositories and platforms that are actively maintained and transparent about updates. Additional safeguards – such as maintaining internal registries of approved components or restricting installations to trusted

---

[22] Machine Learning Principles (National Cyber Security Centre, 2024), https://www.ncsc.gov.uk/sites/default/files/documents/NCSC-Machine-learning-principles.pdf.

[23] For a more in depth discussion of Google and Apple's practices of differential privacy see: 'Learning with Privacy at Scale', Apple Machine Learning Research, accessed 14 May 2026, https://machinelearning.apple.com/research/learning-with-privacy-at-scale; Teng Wang et al., 'Local Differential Privacy for Data Collection and Analysis', *Neurocomputing* 426 (October 2020), https://doi.org/10.1016/j.neucom.2020.09.073; On trade-offs see: *AI Security Concerns in a Nutshell*.

[24] General Data Protection Regulation (GDPR), 2016/679, accessed 15 May 2026, https://gdpr-info.eu/.

[25] *AI Security Concerns in a Nutshell* (Federal Office for Information Security, 2023), https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/KI/Practical_AI-Security_Guide_2023.pdf?__blob=publicationFile&v=5.

[26] 'Google Introduces SLSA Framework', Google Cloud Blog, 30 July 2021, https://cloud.google.com/blog/products/application-development/google-introduces-slsa-framework.

sources – can further reduce the risk of introducing compromised artefacts. This is particularly relevant in ecosystems built around platforms such as GitHub or Hugging Face, where ease of sharing also increases exposure.

A second area of concern is how model artefacts are loaded and executed. In many machine learning environments, model files and configuration formats are not simply static data files; they may also contain instructions that run automatically when models are loaded. This creates a broader software security risk in which seemingly trusted files can execute malicious code.

The AI ecosystem increases this exposure because models are frequently downloaded from public repositories, reused across projects, and integrated through automated workflows. Mitigation therefore focuses on reducing opportunities for unsafe execution and improving checks on external artefacts before they are used. This may include scanning model files and dependencies for suspicious or executable code, using safer loading methods, avoiding higher-risk file formats, and limiting what models are allowed to access when they are loaded. Running models in isolated or sandboxed environments provides an additional layer of protection by restricting what compromised artefacts can access or modify. For example, platforms such as Microsoft's Azure Machine Learning use containerised execution to limit the impact of potentially unsafe artefacts.[27]

A further challenge is the possibility that a model itself has been compromised. Backdoored models may perform well under standard testing while producing manipulated outputs under specific conditions. Because these behaviours are embedded within the model, they are difficult to detect through conventional evaluation.

Addressing this risk requires extending testing beyond standard performance metrics. This may include evaluating models under adversarial conditions, probing for anomalous behaviour, or comparing outputs against independently developed reference models.[28] Red teaming exercises can also help identify vulnerabilities before deployment.[29]

Recent work by Microsoft has also demonstrated promising approaches for detecting backdoored language models before deployment.[30] Rather than relying on knowledge of specific triggers, these techniques scan models for behavioural signatures associated with hidden backdoors, offering a potentially scalable method for screening externally sourced models and model weights.

Across these areas, a common requirement is improved visibility. Organisations need to understand how models are created, modified, and reused over time. Maintaining records of model provenance

---

[27] 'Enterprise Security and Governance - Azure Machine Learning', Microsoft, accessed 14 May 2026, https://learn.microsoft.com/en-us/azure/machine-learning/concept-enterprise-security?view=azureml-api-2.

[28] Huili Chen et al., *DeepInspect: A Black-Box Trojan Detection and Mitigation Framework for Deep Neural Networks*, 2019, 4658–64; Xiaojun Xu et al., 'Detecting AI Trojans Using Meta Neural Analysis', *2021 IEEE Symposium on Security and Privacy (SP)*, May 2021, 103–20, https://doi.org/10.1109/SP40001.2021.00034.h

[29] Vassilev et al., *Adversarial Machine Learning*.

[30] Blake Bullwinkel and Giorgio Severi, 'Detecting Backdoored Language Models at Scale', Microsoft Security Blog, 4 February 2026, https://www.microsoft.com/en-us/security/blog/2026/02/04/detecting-backdoored-language-models-at-scale/

– including training data sources, modification history, and distribution pathways – enables more informed decisions about trust and risk. This is particularly important in environments where pre-trained models are frequently adapted and redistributed, making lineage difficult to track.[31]

Mitigation at the Deployment Level

At the deployment level, mitigation focuses on how AI systems behave once exposed to real-world inputs and users. The primary risk is no longer how systems are built, but how they can be influenced or probed through interaction.

A first priority is managing external inputs. Systems that combine user prompts with retrieved content or system instructions are particularly vulnerable to manipulation. Adversaries can embed malicious instructions within otherwise benign inputs, altering behaviour without compromising the underlying model. Mitigation begins with treating all external inputs as potentially untrusted – applying filtering, anomaly detection, and moderation before inputs reach core system logic. Structuring systems to separate trusted instructions from user-provided content – often referred to as prompt isolation – can further reduce risk. This approach is reflected in systems such as Microsoft Copilot, which uses layered controls to distinguish between instruction types.[32]

A second concern is limiting what systems expose through outputs. Restricting responses to only necessary information reduces the risk of inference attacks, while rate limiting and access controls make large-scale probing more difficult. These measures are commonly implemented in API-based systems, including those provided by organisations such as OpenAI and Anthropic.[33]

Reducing overfitting plays a complementary role. Models that generalise well tend to memorise less, lowering the risk of extraction attacks such as membership inference.[34] However, this does not eliminate the problem: even well-performing models may retain sensitive or atypical data.[35]

Because deployment environments are dynamic, many risks only emerge over time. Continuous monitoring is therefore essential. Logging interactions and analysing usage patterns can help detect suspicious behaviour, such as repeated probing queries or attempts to circumvent safeguards. Cloud providers such as Amazon Web Services increasingly offer tools that support real-time monitoring and the identification of unusual or potentially malicious activity.

---

[31] Beyond individual artefacts, model-level security depends on the integrity of the broader development environment. As the LiteLLM case illustrates, compromises often occur not within the model itself, but in the tooling and pipelines used to build and distribute it. Securing continuous integration and continuous delivery workflows, limiting access to publishing credentials, and verifying build outputs are therefore important components of defence at this stage. Without such controls, attackers may bypass upstream safeguards and introduce malicious functionality into otherwise legitimate artifacts.

[32] 'Data, Privacy, and Security for Microsoft 365 Copilot', Microsoft, accessed 14 May 2026, https://learn.microsoft.com/en-us/microsoft-365/copilot/microsoft-365-copilot-privacy; Xiaoyu Zhang et al., 'JailGuard: A Universal Detection Framework for LLM Prompt-Based Attacks', arXiv:2312.10766, preprint, arXiv, 15 March 2025, https://doi.org/10.48550/arXiv.2312.10766.

[33] https://neuraltrust.ai/blog/rate-limiting-throttling-ai-agents

[34] Practices such as using larger and more diverse datasets, applying regularisation techniques, or incorporating synthetic data can help reduce this exposure. Wenjie Fu et al., 'Membership Inference Attacks against Fine-Tuned Large Language Models via Self-Prompt Calibration', paper presented at The Thirty-eighth Annual Conference on Neural Information Processing Systems, 6 November 2024, https://openreview.net/forum?id=PAWQvrForJ.

[35] Vulnerabilities often come from specific memorised or outlier samples, not just global overfitting, as noted in: Mona Khalil et al., 'Membership Inference Attacks Beyond Overfitting', arXiv:2511.16792, preprint, arXiv, 20 November 2025, https://doi.org/10.48550/arXiv.2511.16792.

Lastly, deployment-level security depends on the broader infrastructure in which models operate. AI systems are embedded within applications that rely on APIs, integration layers, external services, and connected infrastructure. Weaknesses in these components can create alternative pathways for exploitation. Strengthening authentication, access controls, and network protections is therefore a critical part of securing deployed systems. However, these measures often involve trade-offs between security, usability, and system performance, particularly in AI systems designed to operate flexibly across multiple data sources and user environments.

Towards a Layered Approach

A consistent pattern emerges across the data, model, and deployment levels: no single control is sufficient. Risks are distributed throughout the AI supply chain and often only become visible as systems are developed, reused, and deployed.

At the data level, exposure stems from large-scale aggregation and limited visibility into data provenance. At the model level, it arises from reliance on shared components and opaque artefacts. At the deployment level, systems must operate in open environments where adversarial interaction is unavoidable. In each case, mitigation reduces risk but does not eliminate it.

This reflects a broader constraint.

> The features that make modern AI development effective – reuse, scale, and openness – also make it difficult to guarantee the integrity of all components. Eliminating these dependencies is neither feasible nor desirable.

Security must therefore be approached as a ***layered process***. This means combining controls across stages: improving visibility into how components are sourced and used, strengthening safeguards around how they are introduced and executed, and maintaining the ability to monitor and respond to system behaviour in operation.

For policymakers, the implication is clear: AI security cannot be addressed at a single stage of development or deployment. It requires coordinated measures that span data governance, software supply-chain security, and deployment practices. The objective is not to eliminate risk entirely, but to make it more visible, limit its impact, and improve resilience within the system as a whole.

Conclusion

This report has argued that securing artificial intelligence requires a shift in perspective: from focusing on individual models to understanding the broader supply chains through which they are developed, distributed, and deployed. The LiteLLM incident illustrates how adversaries can exploit trusted upstream components to affect multiple systems simultaneously, often without immediate detection.

The specific risks for the data, model, and deployment levels differ, but they share a common feature: they arise from implicit trust in complex and interconnected systems. AI development depends on extensive reuse – of data, code, models, and infrastructure – often across organisational and geographic boundaries. This creates systemic exposure that cannot be addressed through isolated technical fixes.

The mitigation strategies examined in this report point to three core principles.

1

VISIBILITY

Visibility is essential to understanding where dependencies lie and how vulnerabilities may be introduced.

2

CONTROL

Control through access management, verification, and secure configuration reduces the likelihood that compromised components propagate unnoticed.

3

ACCOUNTABILITY

Accountability, supported by testing, monitoring, documentation, and auditability, helps determine how responsibility for security is distributed across developers, providers, and deployers within the AI supply chain, while improving the ability to identify failures and respond effectively when incidents occur.

At the same time, there are clear structural limits. The openness and scalability of the AI ecosystem are not weaknesses to be eliminated, but features that underpin innovation. Efforts to secure AI systems must therefore work with these characteristics, not against them – managing interdependence rather than attempting to remove it.

For policymakers, this has several implications. First, AI security cannot be addressed solely through model-level regulation or post-deployment oversight. It requires attention to upstream practices, including data governance, software supply-chain security, and the integrity of development infrastructure. Second, many of the most significant risks lie outside the direct control of system deployers, highlighting the need for standards, coordination, and shared responsibility among the organisations involved. Third, existing approaches to cybersecurity provide a foundation, but must be adapted to account for the distinctive features of machine learning systems.

For practitioners, the report underscores the need to treat AI systems not as isolated technical artefacts, but as components embedded within broader and evolving environments. Security considerations must therefore be integrated into data collection, model development, system integration, and deployment, rather than addressed only after systems are operational.

Navigating security in the AI supply chain is therefore an ongoing process. As AI systems become more widely deployed and more deeply integrated into critical functions, adversarial incentives will continue to grow. Effective risk management will depend on sustained coordination across technical, organisational, and policy domains – aimed not at eliminating risk entirely, but at making it more visible, constraining its impact, and strengthening resilience over time.

References

AI Security Concerns in a Nutshell. Federal Office for Information Security, 2023. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/KI/Practical_AI-Security_Guide_2023.pdf?__blob=publicationFile&v=5.

Apple Machine Learning Research. 'Learning with Privacy at Scale'. Accessed 14 May 2026. https://machinelearning.apple.com/research/learning-with-privacy-at-scale.

Bernadett-Shapiro, Gabriel and Silas Cutler, 'Silent Brothers: Ollama Hosts Form Anonymous AI Network Beyond Platform Guardrails', SentinelOne Labs, 2025, https://www.sentinelone.com/labs/silent-brothers-ollama-hosts-form-anonymous-ai-network-beyond-platform-guardrails/

Bullwinkel, Blake and Giorgio Severi, 'Detecting Backdoored Language Models at Scale', Microsoft Security Blog, 4 February 2026, https://www.microsoft.com/en-us/security/blog/2026/02/04/detecting-backdoored-language-models-at-scale/

Carlini, Nicholas, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr, 'Poisoning Web-Scale Training Datasets Is Practical', 2024 IEEE Symposium on Security and Privacy (SP), May 2024, https://doi.org/10.1109/SP54263.2024.00179

Chen, Huili, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. DeepInspect: A Black-Box Trojan Detection and Mitigation Framework for Deep Neural Networks. 2019, 4658–64.

Crawford, Kate. *Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence*. New Haven, 2021.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 'ImageNet: A Large-Scale Hierarchical Image Database'. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, 248–55. https://doi.org/10.1109/CVPR.2009.5206848.

Duran, Sergi Gálvez. Mapping Relevant Data Collection Mechanisms for AI Training. OECD Artificial Intelligence Papers 48. OECD, 2025. https://www.oecd.org/content/dam/oecd/en/publications/reports/2025/10/mapping-relevant-data-collection-mechanisms-for-ai-training_62921889/3264cd4c-en.pdf.

Fu, Wenjie, Huandong Wang, Chen Gao, Guanghua Liu, Yong Li, and Tao Jiang. 'Membership Inference Attacks against Fine-Tuned Large Language Models via Self-Prompt Calibration'. Paper presented at The Thirty-eighth Annual Conference on Neural Information Processing Systems. 6 November 2024. https://openreview.net/forum?id=PAWQvrForJ.

General Data Protection Regulation (GDPR), 2016/679. Accessed 15 May 2026. https://gdpr-info.eu/.

Gilbertson, Annie, and Alex Reisner. 'Apple, Nvidia, Anthropic Used Thousands of Swiped YouTube Videos to Train AI'. Tags. Wired, 16 July 2024. https://www.wired.com/story/youtube-training-data-apple-nvidia-anthropic/.

Google Cloud Blog. 'Google Introduces SLSA Framework'. 30 July 2021. https://cloud.google.com/blog/products/application-development/google-introduces-slsa-framework.

Horton, Helena. 'Microsoft Deletes "Teen Girl" AI After It Became a Hitler-Loving Sex Robot Within 24 Hours'. *Telegraph*, 24 March 2016. https://www.telegraph.co.uk/technology/2016/03/24/microsofts-teen-girl-ai-turns-into-a-hitler-loving-sex-robot-wit/.

Hu, Qiang, Xiaofei Xie, Sen Chen, and Lei Ma. 'Large Language Model Supply Chain: Open Problems From the Security Perspective'. arXiv:2411.01604. Preprint, arXiv, 3 November 2024. https://doi.org/10.48550/arXiv.2411.01604.

Khalil, Mona, Alberto Blanco-Justicia, Najeeb Jebreel, and Josep Domingo-Ferrer. 'Membership Inference Attacks Beyond Overfitting'. arXiv:2511.16792. Preprint, arXiv, 20 November 2025. https://doi.org/10.48550/arXiv.2511.16792.

Lee, Peter. 'Learning from Tay's Introduction'. Official Microsoft Blog, 25 March 2016. https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/.

Machine Learning Principles. National Cyber Security Centre, 2024. https://www.ncsc.gov.uk/sites/default/files/documents/NCSC-Machine-learning-principles.pdf.

Microsoft. 'Data, Privacy, and Security for Microsoft 365 Copilot'. Accessed 14 May 2026. https://learn.microsoft.com/en-us/microsoft-365/copilot/microsoft-365-copilot-privacy.

Microsoft. 'Enterprise Security and Governance - Azure Machine Learning'. Accessed 14 May 2026. https://learn.microsoft.com/en-us/azure/machine-learning/concept-enterprise-security?view=azureml-api-2.

MITRE ATLASTM. 'VirusTotal Poisoning'. Accessed 14 May 2026. https://atlas.mitre.org/studies/AML.CS0002.

Pushkarna, Mahima, Andrew Zaldivar, and Oddur Kjartansson. 'Data Cards: Purposeful and Transparent Dataset Documentation for Responsible AI'. arXiv:2204.01075. Preprint, arXiv, 3 April 2022. https://doi.org/10.48550/arXiv.2204.01075.

Roush, Ty. 'Reddit Accuses Anthropic Of Taking User Data Without Consent'. *Forbes*, 4 June 2025. https://www.forbes.com/sites/tylerroush/2025/06/04/reddit-claims-anthropic-trained-ai-models-using-its-users-data-without-consent/.

Russakovsky, Olga, Jia Deng, Hao Su, et al. 'ImageNet Large Scale Visual Recognition Challenge'. *International Journal of Computer Vision* 115, no. 3 (2015): 211–52. https://doi.org/10.1007/s11263-015-0816-y.

Russell, Stuart, and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th edn. Pearson, 2021.

Sherman, Justin. Securing Data in the AI Supply Chain. Atlantic Council, 2025. https://www.atlanticcouncil.org/in-depth-research-reports/issue-brief/securing-data-in-the-ai-supply-chain/.

Souly, Alexandra, Javier Rando, Ed Chapman, et al. 'Poisoning Attacks on LLMs Require a Near-Constant Number of Poison Samples'. arXiv:2510.07192. Preprint, arXiv, 8 October 2025. https://doi.org/10.48550/arXiv.2510.07192.

Steinhardt, Jacob, Pang Wei W. Koh, and Percy S. Liang. 'Certified Defenses for Data Poisoning Attacks'. *Advances in Neural Information Processing Systems* 30 (2017). https://proceedings.neurips.cc/paper/2017/hash/9d7311ba459f9e45ed746755a32dcd11-Abstract.html.

Tramèr, Florian, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. 'Stealing Machine Learning Models via Prediction APIs'. arXiv, 9 September 2016. https://arxiv.org/abs/1609.02943v2.

Trend Micro, 'Preventing Zero-Click AI Threats: Insights from EchoLeak,' Accessed 15 July 2026, https://www.trendmicro.com/en_us/research/25/g/preventing-zero-click-ai-threats-insights-from-echoleak.html.

Vassilev, Apostol, Alina Oprea, Alie Fordyce, Hyrum Anderson, Xander Davies, and Maia Hamin. *Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations*. NIST Artificial Intelligence (AI) 100-2 E2025. National Institute of Standards and Technology, 2025. https://doi.org/10.6028/NIST.AI.100-2e2025.

Venkatesan, Sridhar, Harshvardhan Sikka, Rauf Izmailov, Ritu Chadha, Alina Oprea, and Michael J. de Lucia. 'Poisoning Attacks and Data Sanitization Mitigations for Machine Learning Models in Network Intrusion Detection Systems'. *MILCOM 2021 - 2021 IEEE Military Communications Conference with MILCOM*, November 2021, 874–79. https://doi.org/10.1109/MILCOM52596.2021.9652916.

Wang, Teng, Jun Zhao, Zhi Hu, Xinyu Yang, Xuebin Ren, and Kwok-Yan Lam. 'Local Differential Privacy for Data Collection and Analysis'. *Neurocomputing* 426 (October 2020). https://doi.org/10.1016/j.neucom.2020.09.073.

Xu, Xiaojun, Qi Wang, Huichen Li, Nikita Borisov, Carl A. Gunter, and Bo Li. 'Detecting AI Trojans Using Meta Neural Analysis'. 2021 IEEE Symposium on Security and Privacy (SP), May 2021, 103–20. https://doi.org/10.1109/SP40001.2021.00034.

Zhang, Xiaoyu, Cen Zhang, Tianlin Li, et al. 'JailGuard: A Universal Detection Framework for LLM Prompt-Based Attacks'. arXiv:2312.10766. Preprint, arXiv, 15 March 2025. https://doi.org/10.48550/arXiv.2312.10766.

virtual routes

For more information, please visit: **www.virtual-routes.org**

If you have any further queries, questions, or concerns, feel free to reach out via email at:
contact@virtual-routes.org