# AI-Specific Attack Vectors: Prompt Injection, Data Poisoning, and Model Extraction

Adam Khalil

June 2025

Explore the key vulnerabilities, techniques, and defense strategies surrounding AI attack vectors—data poisoning, prompt injection, and model extraction—in this in-depth guide designed for security-minded researchers, developers, and AI professionals.

## Introduction

According to Gartner, data shows a 187% growth in enterprise AI adoption from 2023 to 2025, but only a 43% increase in security spending. This merits concern because AI-related breaches are costly. Seventy-three percent of enterprises experienced at least one AI security incident in the past year, at an average cost of $4.8 million per breach. In one case, attackers used AI-generated voice deepfakes to steal around $18.5 million in a Hong Kong crypto heist.

Unfortunately, the problem is growing. Security analysts warn that AI-tailored attacks will continue to rise. IBM notes that AI breaches take about 290 days to detect (vs. 207 for traditional breaches), highlighting stealth. Projections of total cybercrime reach around $23 trillion by 2027, suggesting that AI-enabled attacks could drive multi-trillion-dollar losses without stronger AI defenses. The World Economic Forum warns that unchecked AI security failures could cost the global economy approximately $5.7 trillion by 2030.

## Specific Attack Vectors

As generative AI and large language models (LLMs) are integrated into more applications, attackers have developed new tactics to exploit them. Three key AI-specific attack types are prompt injection, data poisoning, and model extraction. Each targets a different stage of the AI pipeline, but can have serious consequences for security, privacy, and trust.

### Prompt Injection Attacks

A prompt injection occurs when an attacker tricks an AI system by inserting hidden instructions into the input prompt. In plain terms, the attacker secretly adds commands to what appears to be a standard user input, causing the AI to disregard its original rules and perform an action it shouldn't. Think of it like sneaking a cheat code.

In practice, attackers can lure an AI-powered assistant into disclosing sensitive data or performing unauthorized actions. For instance, a hacker might craft a query to a virtual assistant that causes it to email private documents to an outside address. Other experiments have shown that attackers can insert malicious

links or code snippets on websites that AI models later scrape, causing them to advise users to click on phishing links inadvertently.

Prompt injections can lead to data leaks, misinformation, and unauthorized actions. An AI might be tricked into revealing confidential prompts, internal logic, or private data stored behind it. It could generate misleading or harmful text, spread false information, or even instruct users to visit malicious sites. In enterprise settings, an attacker could manipulate an AI-driven system (like an email assistant) to act against policies, such as bypassing content filters or sending insider data out of the organization. With tools like AlphaAI's GenAI Firewall and prompt-layer protection, Auxin could help filter harmful inputs and prevent LLM misuse across enterprise workflows. You can learn more about this type of attack here.

## Data Poisoning Attacks

Data poisoning occurs when an attacker contaminates the training data of an AI system, thereby corrupting its behavior. In simpler terms, it's like sneaking poisonous ingredients into the "food" a model eats during training. By feeding the model bad or misleading examples, the attacker causes it to learn incorrect or biased rules.

Modern AI models are trained on large datasets. If an adversary can insert or alter some of this data before or during training, they can steer the model's future outputs. For instance, in a spam filter training process, the attacker might inject many spam emails labeled as "ham" (not spam). When the model is trained, it learns incorrect patterns and will then misclassify spam as legitimate email.

Poisoning can silently undermine an AI system. It may lead to inaccurate predictions (e.g., a healthcare model overlooking certain diseases) or biased outcomes (e.g., skewed loan approvals). At scale, poisoning can erode trust: users receive incorrect answers and lose faith in the AI. It can also open backdoors: a poisoned model might behave normally until it sees a special trigger, then perform a secret malicious task (a "backdoor" attack).

In sensitive domains such as autonomous vehicles or medical diagnosis, even minor errors can lead to accidents, financial loss, or legal trouble. For example, if a self-driving car's vision model is poisoned, it might ignore stop signs or misinterpret road markers, posing real hazards. Auxin's AlphaScale and AlphaCloud platforms offer secure CI/CD scanning, data integrity checks, and cloud-based monitoring to detect poisoned inputs early in the training pipeline. Moreover, industries bound by regulation (healthcare, finance) could face compliance breaches if a model trained on poisoned data makes a harmful error.

## Model Extraction Attacks

A model extraction (or "model stealing") attack is when an adversary copies or recreates a proprietary AI model by querying it repeatedly. In simpler terms, the attacker treats the model like a black-box: they submit many inputs to its public interface (API or chatbot) and observe the outputs, then use that "question-and-answer" data to train their duplicate. Model extraction is like ordering every dish on a restaurant's menu—over and over—not to eat them, but to figure out the secret recipe.

Model extraction only requires the attacker to interact with the model a sufficient number of times. For example, suppose a company offers a chatbot API. The attacker systematically feeds it carefully chosen prompts and records the responses. Over thousands of queries, they build a synthetic "dataset" of inputs and outputs. Then they train their model on this synthetic data. The result is a copycat model that behaves almost identically to the original.

One example of this is in 2021, when a team of researchers set out to clone OpenAI's GPT-3, a powerful language model that was only accessible via a paid API. Since they couldn't access the actual model weights, they used model extraction. They sent thousands of prompts to GPT-3, then recorded the responses, treating each prompt-response pair as a data point. Eventually, they used that collected data to train a new model, GPT-J, an open-source alternative. The latest model closely mimicked GPT-3's behavior, despite never having access to the original code or training data. While not a perfect copy, it was capable of producing text of similar quality and was open to the public, unlike GPT-3. This showed that any company exposing a valuable AI model through a public API is at risk. A competitor (or attacker) could clone their model simply by observing enough inputs and outputs, thereby stealing intellectual property and bypassing years of development and cost. By applying API rate limiting, output controls, and identity monitoring through AlphaID and AlphaOpSec, Auxin can provide organizations with a defense-in-depth strategy against model theft.

# How to Defend Your AI Systems

While these attacks differ in technique, they all share a common goal: to manipulate or steal from AI systems, causing harm, gaining access, or retrieving valuable assets. Fortunately, with the proper precautions, organizations can significantly reduce their risk.

## Combatting Prompt Injection Attacks

A great place to start is to validate and sanitize inputs. Check user prompts for hidden commands or unusual content (e.g., overly long text or system-like instructions) before feeding them to the AI. Keep detailed logs of AI queries and monitor for spikes or unusual patterns that may indicate an attack. Set up alerts so that security teams can quickly investigate any unusual behavior.

Additionally, ensure that AI privileges are limited and that humans are involved in the process. Apply the principle of least privilege: only give the AI access to the functions it needs. For any sensitive action (accessing private data, changing settings, etc.), require a human to review or approve the AI's output. Additionally, it is essential to keep your systems up to date. Regularly patch AI models and tools (newer versions often have improved safety) and train your staff on spotting suspicious prompts. AlphaAI from Auxin Security features advanced prompt filtering, role-based output controls, and secure LLM routing to safeguard AI applications against manipulation and misuse.

## Combatting Data Poisoning

Ensure that you validate and sanitize training data. Carefully check incoming data for quality and consistency before using it. Use schema checks, filters, or anomaly detection tools to identify and remove malformed or outlier records from the training data. Good data hygiene keeps hidden "poison" out of your models.

Next, utilize training techniques that reduce the model's sensitivity to poisoned samples. For example, adversarial training or ensemble methods can help the model learn to ignore outliers. Also, regularly test models on clean benchmark data to ensure they still perform correctly. Track the origin of your training data and watch for any unusual trends. If a model suddenly misbehaves (e.g., accuracy drops or strange

predictions), investigate the training data for possible tampering. Detecting unexpected changes early can prevent poisoned data from corrupting the model.

Moreover, lock down your training data sets so that only authorized team members can modify them. Utilize robust access controls (roles, permissions) and encrypt data both at rest and in transit. This prevents attackers or careless insiders from injecting insufficient data.

Finally, periodically retrain models with fresh, verified datasets and reevaluate their performance. Treat this as a routine "health check." Testing your AI against known clean data will help you spot any lingering effects of poisoning attacks. With AlphaScale's CI/CD scanning and AlphaCloud's data posture monitoring, Auxin can enable secure model development by catching potential security issues before they reach production.

## Combatting Model Extraction

A significant first step is to limit query rates. Throttle the number of requests each user or IP can make. Enforcing strict rate limits prevents an attacker from quickly "leaking" the model by massive querying. Combine this with CAPTCHA or other checks if abuse is detected. Also, keep detailed logs of all model queries. Watch for suspicious patterns, such as a single user making multiple requests that are slightly different from each other. An alert or block on such behavior can stop a model-stealing attempt in its tracks. Additionally, only return the information users need. For example, avoid giving full probability scores or extra metadata. You can also add a small amount of random noise or occasionally provide a harmless dummy answer to make exact cloning more difficult.

Other considerations include techniques such as watermarking or encrypting model components. Watermarking (e.g., having the model sometimes output a hidden tag) and model obfuscation make stolen copies less valuable. Together with the above steps, this helps safeguard your AI's intellectual property. Auxin Security's platforms help reduce the risk of model extraction through anomaly detection, usage monitoring, and output controls that make AI systems more difficult to reverse-engineer.

# So, where are we going with this

By 2025, over 70% of organizations are expected to deploy AI in some form. Yet, most lack the tools to defend against AI-specific attacks, such as prompt injections, data poisoning, and model theft.

As the AI market is projected to grow to an estimated $1.3 trillion by 2030, securing the entire AI lifecycle becomes increasingly critical. Auxin Security supports this evolution by delivering advanced tools and services that help organizations build, monitor, and protect AI systems from code to production.