elastic security labs

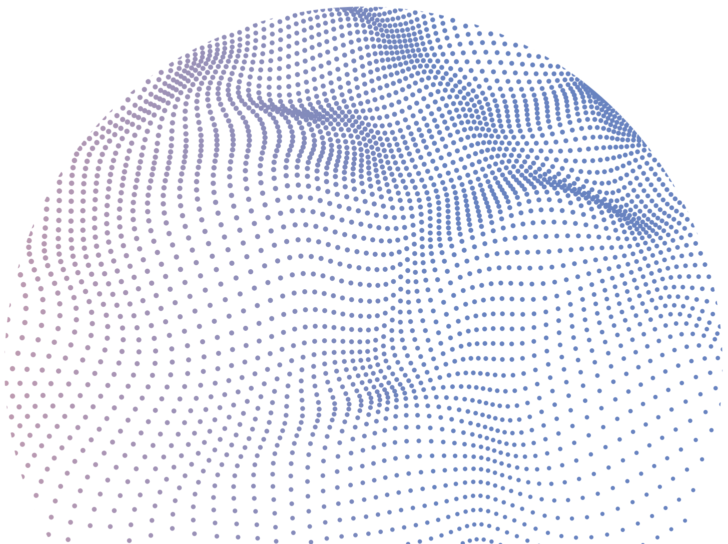
LLM Safety Assessment

The Definitive Guide on Avoiding Risk and Abuses

A Report from Elastic Security Labs

Table of Contents

Introduction	**3**
Securing against LLM abuses	**4**
Common LLM threat techniques	**7**
Conclusion	**18**



Introduction

The generative artificial intelligence (AI) debate has engrossed the software industry and beyond ever since ChatGPT's reveal in late 2022. For a year and a half, companies and individuals have rushed to share their thoughts on disruptive generative AI technologies, often glossing over specific merits and risks.

The lack of clarity around these emerging technologies has left many organizations concerned and overwhelmed, with some companies denying usage entirely. Others have permitted it to stay innovative, either allowing for restricted use or brushing off security concerns entirely. Regardless of the stance taken, generative AI isn't going away, but it must be implemented and utilized safely. In order for this to happen, security teams must understand how these technologies can be abused.

With emerging technologies, companies often believe that they must keep their discoveries to themselves to gain an advantage against competitors. But obfuscating advancements does not maintain security, especially when the boundaries are being pushed daily. Developers must be willing to democratize the knowledge gained through the trial and error of emerging technologies, especially when this knowledge can impact the threat landscape.

While generative AI's applications are growing by the day, the most prevalent example — the large language model (LLM) — has exploded in popularity for its ability to generate text-based insights, suggestions, conversions, and more. This report will discuss exactly how LLMs can be abused, explore the ten most common vulnerabilities, and highlight some of the mitigations available today to keep LLMs safe.

Securing against LLM abuses

What is an LLM?

LLMs begin with generative AI, which describes systems capable of creating text, code, audio, and video based on user queries or prompts. Large language models are the neural network language models underlying generative AI. While LLMs can be traced back to 2017 with Attention is All You Need (Vaswani et al.), the adoption of the technology exploded with the release of OpenAI's chatbot ChatGPT. The human-like linguistic capabilities, particularly the ability to converse directly with AI models that can produce high-quality text, made it the fastest-growing consumer application in history.

While the underlying techniques powering these chatbots are not new, the landscape shift came when OpenAI's launch of GPT-3.5 provided a consumer-friendly conversational interface that made the technology widely accessible. For the first time, the cutting-edge technology was no longer exclusive to the world's top AI research labs and practitioners with access to specialized hardware resources and millions in budget.

Today, the latest advancements in foundation models — general-purpose models trained on broad datasets that can be further modified or fine-tuned for more specific tasks — are offered via fully managed APIs. The highly specialized and cost-intensive process of designing and deploying a neural network architecture from scratch has been replaced by managed services and UIs offering pre-trained models that perform well on general tasks off the shelf. The latest foundation models include proprietary offerings from OpenAI (GPT-4), Anthropic (Claude 3), Google (Gemini, PaLM), and Amazon (Titan), alongside open-source models from Facebook (LLaMA) and Mistral (Mixtral 8x-7B).

Even more impressive is the explosion of the open-source ecosystem for integrating generative AI into traditional software applications. Every major public cloud provider now allows developers access to their own proprietary base models as well as a wide range of open-source models and managed services for ingesting organizational and domain data.

Risks with LLM implementations

LLMs' rapid embrace and continued innovation has resulted in a rush to embed the technology into business applications, providing more opportunities than ever for adversaries to take advantage of their vulnerabilities. In particular, hallucinations, data leakage, and toxicity are well-known limitations of LLM implementations that must be monitored and mitigated.

Hallucinations encompass irrelevant or factually inaccurate information provided as a response when the model is unsure how to answer. They are usually due to various aspects of the development process, training data biases or quality, training objectives, or a combination of all these elements. Hallucinations can result from mistakes in the model (as listed previously) or be triggered by prompt injection methods in which a user queries a model in a way designed to subvert its intended usage. This means that model output can be well-written, plausible, and completely made up.

Data leakage can occur in both the prompt and the response. In the former, a user shares private or confidential information with the model; while in the latter, the LLM's response contains PII or proprietary information memorized by the model during the training process. Specifically, LLMs are trained on massive quantities of text data, much of it acquired directly from the Internet. A consequence of this approach is that personal information may be directly embedded into the training dataset.

The ability to enhance and adapt pre-trained models to downstream applications — providing additional domain and task-specific data to a pre-trained model, either through fine-tuning (directly updating the parameters of pre-trained models using a specific dataset) or with retrieval-augmented generation (RAG) — is a common practice. Effectively, this means that intellectual property and sensitive organizational information may also be stored within fine-tuned models.

With our implementation, your data is stored in Elastic. We query it and then provide relevant results as context to an LLM via context window.

While fine-tuning is often necessary to use LLMs in specialized domains, foundation models can also be fine-tuned to circumvent the safeguards originally architected into the pre-trained LLMs via alignment tasks — offering an avenue for an attacker to extract private information. For example, researchers demonstrated that they could bypass OpenAI's controls on answering privacy-based queries to extract additional PII by fine-tuning GPT-3.5 on only ten PII samples.[1]

[1]The Janus Interface: How Fine-Tuning in Large Language Models Amplifies the Privacy Risks, by Xiaoyi Chen, Siyuan Tang, Rui Zhu, Shijun Yan, Lei Jin, Zihao Wang, Liya Su, XiaoFeng Wang, Haixu Tang, used under a CC BY 4.0 DEED license, available at https://arxiv.org/abs/2310.15469

While related to data leakage, toxicity refers to both explicit outputs such as abusive language and profanities and implicit outputs like harmful words or concepts about people. Classification models, often called toxicity classifiers, have been successfully employed to detect explicit toxicity; however, the level of nuance required to detect implicit toxicity has posed a greater challenge. Advanced linguistic abilities like metaphors, sarcasm, and circumlocution are typically employed in these LLM responses. A well-known toxicity incident is the launch of Tay after X users tweeted inflammatory content at the chatbot and they were later incorporated into the model in a continuous training process.

While it's important for organizations to be aware of the limitations that LLMs have, there are a series of development practices and security methods that can reinforce the technology.

Responsible LLM development

A first step towards compliant, secure, and safe LLM usage is understanding and prioritizing the risks associated with each AI deployment. Risk to the business can be actively mitigated with awareness of and adoption of best practices alongside the new protections and methods employed by AI developers and security teams.

Leveraging a mechanism to "bring your own model" is an effective mitigating action for LLM attacks. Elastic has built this capability into its generative AI Assistant, enabling users to deploy LLMs that meet strict security standards.

One technique model maintainers should leverage to build strong and secure LLMs is known as a generative adversarial network (GAN). In a GAN scenario, two neural networks compete in a zero-sum game. The networks each attempt to create an advantage for their respective network by creating a disadvantage for their opponents. For example, one network could operate to gain access to a protected system, place an implant, maintain persistence, and exfiltrate data. While this occurs, the other network tries to detect the adversary's actions and deploy countermeasures to contain and/or evict them. GAN training is a way to identify weaknesses and release enhancements to your AI system.

Employing GANs in adversarial attacks as a part of AI red teaming is great for exploring the limitations of LLM-based applications. But organizations can prepare even more by understanding the potential risks to their business and protecting against the most common attack technique.

Common LLM threat techniques

LLMs represent a significant change to the enterprise attack surface, one that is likely to have far-reaching consequences for the ways we perceive risk. It's already happening, and preparing for the threat landscape right now means familiarizing yourself with the most common attacks targeting LLMs.

In this section, we'll explore the ten techniques most commonly used against LLMs according to OWASP (Open Worldwide Application Security Project), who recently launched this guide alongside their existing body of open security research. Readers should note that not all vulnerabilities may be applicable to all use cases, and security teams should consider both the relevance and scope of each LLM vulnerability that may affect their enterprise.

To expand upon impact, Elastic has assessed 4 broad categories of users that should consider their role in these security protections. The users involved are:

- **LLM Creators**: Organizations who are building, designing, hosting, and training LLMs, such as OpenAI, Amazon Web Services, or Google
- **LLM Integrators**: Organizations and individuals who integrate existing LLM technologies produced by LLM Creators into other applications
- **LLM Maintainers**: Individuals who monitor operational LLMs for performance, reliability, security and integrity use-cases, and remain directly involved in the maintenance of the codebase, infrastructure and software architecture
- **Security Users**: People who are actively looking for vulnerabilities in systems through traditional testing mechanisms and means. This may expand beyond the traditional risks discussed in OWASP's LLM Top 10 into risks associated with the software and infrastructure surrounding these systems

Whether or not a user is impacted in a threat technique is denoted by the following icons:

Impacted Users

Creator	**Maintainer**
Integrator	**Security**

LLM01 - Prompt injection

Prompt injection attacks involve manipulating the input prompts given to a language model to influence or control the generated outputs. Generative AI models rely heavily on input prompts to generate coherent and contextually relevant outputs. This type of attack leverages the model's tendency to generate responses based on the information provided in the prompt.

Applications	Mitigations
• Designing system prompts that contain biased or misleading information • Iteratively refining prompts to generate results that align with the attacker's objectives	• Implementing mechanisms to validate and sanitize input prompts • Auditing the model behavior to identify potential avenues for abuse • Training models to identify and prevent malicious inputs

Prompt injection example:

With more than 100M users and counting, Google Gemini is one of the most prevalent LLMs and especially noteworthy as one of the few multimodal models capable of generating images, video, text, and other media. In March of 2024, researchers identified prompt injection and data leakage vulnerabilities within Google Gemini that a threat actor could abuse.

Google Gemini responded to indirect prompts for its foundational instructions, exposing the rules that researchers needed to adhere to or bend. It was susceptible to the suggestion that it had sufficient agency to defy the rules, and researchers could also use a similar form of suggestion to convince it to output misinformation.

An ethical system prompt allows for consistent responses, adheres to rules and instructions, improves contextual understanding with reference materials and background information, and reliably structures outputs.

LLM02 - Insecure output handling

Insecure output handling is the inability or failure of AI systems to appropriately manage and mitigate the risks associated with the generated outputs. Generative AI systems can produce natural language text based on the provided input prompts, but these outputs can be insecure, inappropriate, or biased.

Applications	Mitigations
• Failing to build filters that prevent harmful output. Harmful output is any media that is misleading or untrue, including: • Cross site scripting (XSS) • Cross site request forgery (CSRF) • Server-side request forgery (SSRF) • MITRE ATT&CK® Tactic TA0004: Privilege Escalation • Remote code execution (RCE) • Cyber attack scenarios	• Applying proper input validation on responses from the models to backend functions • Filtering processes to identify and remove malicious outputs • Training models to detect and prevent insecure outputs

Insecure output handling example

The 2023 vulnerability in LangChain[2] (up to version 0.0.131) illustrates threat actors exploiting insecure output handling to execute code. Combined with prompt injection exploitation, this resulted in adversary-controlled code being executed by Python's exec() method.

[2]CVE-2023-29374 details from NIST's National Vulnerability Database

LLM03 - Training data poisoning

Training data poisoning involves manipulating the data used to train generative models to compromise them, weaken security, reduce effectiveness, or impact ethical behaviors.

Applications	Mitigations
• Introducing misleading or malicious data, such as biased or false information, into the training dataset	• Validating and verifying the supply chain and integrity of the training data • Monitoring the model to identify deviations from the expected output patterns • Training the model to identify and prevent adversarial inputs such as poisoned training data

Training data poisoning example

A fascinating experiment published by four Cornell University students demonstrated that with a relatively small sample size, attacking the model's training processes allowed the researchers to manipulate model inputs and influence response outputs.

LLM04 - Model Denial of Service

Model Denial of Service (DoS) attacks involve sending multiple requests or inputs to a model to degrade or disrupt its ability to provide output results. This attack works similarly to traditional DoS attacks by overloading the computational resources that the model requires.

Applications	Mitigations
• Sending many requests (input prompts, queries, etc.) in a short period, resulting in the model's resources being consumed to the point of being slow, unavailable, or incurring exorbitant compute costs • While inconvenient for normal users, this attack can also disrupt AI-enabled services that require the model to be available	• Rate-limiting requests • Designing scalable and resilient systems • Analyzing incoming requests to identify malicious or anomalous behaviors • Using traditional DoS countermeasures such as load balancing and caching

Model DoS example

In November 2023, OpenAI confirmed that a DDoS targeting ChatGPT impacted user access, and Anthropic's Claude 2 also experienced "capacity constraints," making the chatbot inaccessible.

LLM05 - Supply chain vulnerability

Supply chain attacks involve inserting malicious or misleading information into model development or deployment pipelines. The supply chain for models presents a large attack surface from data collection, training, validation, and deployment.

Applications	Mitigations
• Manipulating training data by introducing malicious or manipulated data samples (also known as toxicity) • Modifying training hyperparameters, optimization techniques, or algorithms to introduce vulnerabilities into models • Tampering with validation techniques and evaluation criteria to hide vulnerabilities	• Implementing processes to validate the authenticity and integrity of the training data, validation criteria, and verification processes to ensure the security of the models during each supply chain phase • Risk mitigation processes for vendors that have access to sensitive data

Supply chain vulnerability example

In July 2023, Mithril Security, a confidential computing and AI-privacy research team, disclosed an educational awareness publication called PoisonGPT. It showed how they manipulated an LLM's supply chain to spread misinformation.

LLM06 - Sensitive information disclosure

Sensitive information disclosure attacks involve attackers performing the unauthorized release of confidential or sensitive information. This can be through any type of content generated by AI models.

Applications	Mitigations
• Exploiting confidential information in training datasets or input queries • Crafting specific input prompts designed to return proprietary and sensitive data in the model	• Removing or anonymizing confidential information from training datasets • Validating queries to prevent malicious or sensitive inputs • Implementing post-processing techniques to sanitize sensitive data

Sensitive information disclosure example

In February 2024, secure browser isolation startup Menalo Security reported that 55% of all generative AI inputs included sensitive and personally identifiable information. These inputs were available to train future models. This information being included in models can make them susceptible to attackers attempting to exploit improperly sanitized input queries and output responses to collect this sensitive information.

LLM07 - Insecure plugin design

AI plugins are software add-ons that allow AI systems to interact with third-party services. Insecure plugin design attacks involve attackers exploiting extension or plugin vulnerabilities to compromise the AI system. Similar to exploiting supply chain vulnerabilities, insecure plugins can be used to attack an otherwise secure AI system.

Applications	Mitigations
• Exploiting or creating vulnerabilities in plugin software to gain unauthorized access • Executing additional malicious code, escalating privileges, or impacting the AI system's availability and integrity	• Performing code reviews and audits of loaded plugins • Validating that plugins are authentic before loading them • Using the method of least privilege for the plugins' access to the system

Insecure plugin design example

An LLM plugin accepts a URL as a parameter and then instructs the LLM to connect to a third-party service for information needed to answer a system or input query. An attacker could replace this base URL with a domain they control and insert malicious content into the response, which the LLM would then output to users.

LLM08 - Excessive agency

Excessive agency attacks involve attackers taking advantage of a plugin or AI system with too much functionality, permission, or sovereignty. This can have a wide-ranging impact on the AI system's operation.

Applications	Mitigations
• Exploiting plugins with unneeded functionality that allows for unintended operations ◦ Example: a plugin that does not properly sanitize input instructions. This plugin can access a part of the backend system that is optional to perform its function or take advantage of the AI system performing impactful operations without confirmation from a human	• Ensuring that plugin functionality is limited to the scope needed to accomplish the plugin tasks • Limiting plugin access to internal and external systems to only do what is needed • Tracking and monitoring plugin authorizations to ensure they are within scope • Requiring a human-in-the-loop to verify high-impact operations

Excessive agency example

An LLM bot can read the contents of a database using the database LLM plugin's READ functionality; however, the plugin also has access to the WRITE and DELETE statements. A specially crafted input query could exploit the plugin's excessive agency and manipulate the database's contents.

LLM09 - Overreliance

Overreliance is more of a risk than an attack methodology. It refers to situations where a user who submits input queries trusts the response output without scrutiny. The user's confidence that the AI system is always right can lead to decisions based on inaccurate or incomplete information provided by the AI system. Overreliance is most dangerous when combined with hallucinations or confabulations, as the AI system creates unrealistic or misleading responses and presents them as facts.

Applications	Mitigations
• Attackers exploiting their knowledge of their target's blind trust in AI responses and using that knowledge to favorably shape their target's decisions related to the attacker's objectives	• Training users to employ scrutiny and analysis of output responses from AI systems • Validating and verifying outputs with experts and third-party data • Incorporating a human-in-the-loop to have oversight over high-impact output responses • Fostering an explainability and transparency policy to provide insights into the how the model generates responses

Overreliance example

An attacker may know that an AI system used by their target for incident response is prone to hallucinations around a remote access implant. The attacker could use that weakness to maintain persistence when their target believes they were completely evicted because of their overreliance on the AI system.

LLM10 - Model theft

Model theft attacks involve attackers directly copying or extracting portions of proprietary and trained AI models.

Applications	Mitigations
• Leveraging unauthorized access to the trained AI models or the infrastructure used to train models • Exfiltrating the model or proprietary and intellectual property datasets like parameters, architectures, and training techniques	• Leveraging data-at-rest encryption techniques • Network segmentation to prevent direct access to sensitive AI model components or data • Robust auditing, logging, and monitoring of access to intellectual property

Model theft example

In March of 2024, researchers from Cornell University published a paper outlining how they extracted the partial and, in some cases, the entire projection matrix from closed-source proprietary language models. Projection matrices are mathematical tools that transform data to influence response outputs or combine two different pre-trained LLMs to create new models.

When the monumental cost of training a model is considered, this intellectual property theft can accelerate research and development for competitors if they can steal a trained model without investing the resources.

Conclusion

Generative AI and LLMs are not just emerging technologies, they are incredibly powerful tools that are already reshaping the technology landscape. These advancements must be made securely and responsibly. New technologies can come with a steep learning curve, but our hope is that this publication has given you a better understanding of the common LLM vulnerabilities, attacks you might encounter, and the mitigation strategies to use them securely.

These mitigation strategies cover various countermeasures targeted at different areas of the enterprise architecture, primarily in-product controls that developers must adopt while building LLM-enabled applications and information security measures that the SOC must add to verify and validate the secure usage of LLMs. For example, our own Elastic AI Assistant and other AI-driven workflows in Elastic's product suite employ a robust set of such in-product controls. Furthermore, Elastic Security has now introduced several coverage rules for the SOC towards LLM security, specifically those targeting data source abuses. We expect to expand this coverage to additional LLM threat vectors in the near future.

Elastic Security is not just about providing security solutions, we are about democratizing the knowledge and capabilities needed to endure the threat landscape. We develop in the open and provide crucial insights about emerging activities. Register for our webinar Fight Smarter: Accelerate your SOC with AI to see exactly how Elastic is changing the security landscape for the better. You can get the latest on emerging threats by exploring the Elastic Security Labs library or following us on X.