Large Language Model (LLM) Threats Taxonomy

AI Controls Framework Working Group

CSA cloud security alliance®

The permanent and official location for the AI Controls Framework Working Group is https://cloudsecurityalliance.org/research/working-groups/ai-controls

Acknowledgments

Lead Authors

Siah Burke
Marco Capotondi
Daniele Catteddu
Ken Huang

Contributors

Marina Bregkou
Sanitra S. Angram
Vidya Balasubramanian
Avishay Bar
Monica Chakraborty
Anton Chuvakin
Ricardo Ferreira
Alessandro Greco
Krystal Jackson
Gian Kapoor
Kushal Kumar
Ankita Kumari
Yutao Ma
Danny Manimbo
Vishwas Manral
Jesus Luna
Michael Roza
Lars Ruddigheit
Dor Sarig
Amit Sharma
Rakesh Sharma
Kurt Seifried
Caleb Sima
Eric Tierling
Jennifer Toren
Rob van der Veer
Ashish Vashishtha
Sounil Yu
Dennis Xu

Reviewers

Phil Alger
Ilango Allikuzhi
Bakr Abdouh
Vinay Bansal
Vijay Bolina
Brian Brinkley
Anupam Chatterjee
Jason Clinton
Alan Curran
Sandy Dunn
David Gee
Zack Hamilton
Vic Hargrave
Jerry Huang
Rajesh Kamble
Gian Kapoor
Rico Komenda
Vani Mittal
Jason Morton
Ameya Naik
Gabriel Nwajiaku
Meghana Parwate
Prabal Pathak
Ruchir Patwa
Brian Pendleton
Kunal Pradhan
Dr. Matt Roldan
Omar Santos
Dr. Joshua Scarpino
Natalia Semenova
Bhuvaneswari Selvadurai
Jamillah Shakoor
Tal Shapira
Akram Sheriff
Srinivas Tatipamula
Maria (MJ) Schwenger
Mahmoud Zamani
Raphael Zimme

Table of Contents

Objectives and Scope

This document was authored by the Cloud Security Alliance (CSA) Artificial Intelligence (AI) Controls Framework Working Group, within the context of the CSA AI Safety Initiative. It establishes a common taxonomy and definitions for key terms related to risk scenarios and threats to Large Language Models (LLMs). The goal is to provide a shared language and conceptual framework to facilitate communication and alignment within the Industry and to support additional research within the context of the CSA AI Safety Initiative. More specifically, these definitions and taxonomy are intended to assist the CSA AI Control Working Group and the CSA AI Technology and Risk Working Group in their ongoing efforts.

In this effort, we focus on the definition of the following elements (See Figure 1):

- LLM Assets
- LLM-Service Lifecycle
- LLM-Service Impact Categories
- LLM-Service Threat Categories

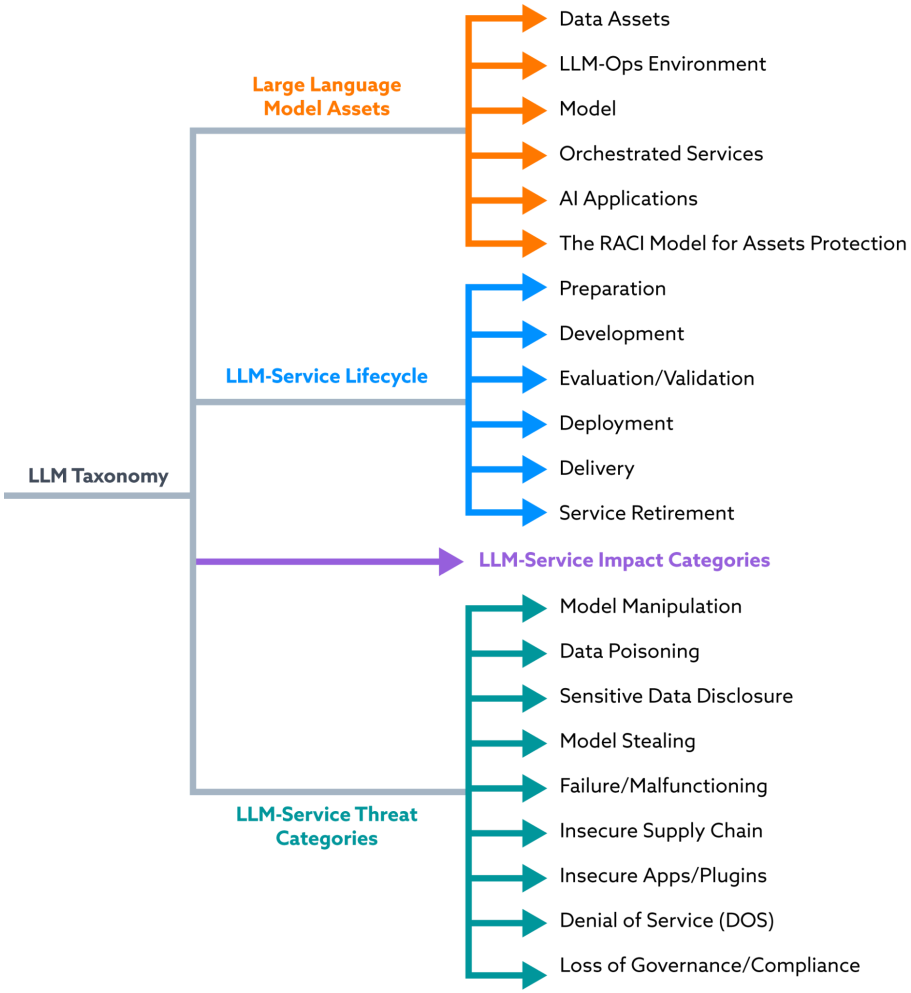

Figure 1: CSA LLM Threat Taxonomy

These definitions and taxonomy reflect an extensive review of the available literature, as well as meetings and discussions among Working Group members and co-chairs. Through this collaborative exercise, a strong consensus emerged, establishing a foundational set of common terminologies guiding our collective efforts.

This document draws inspiration from numerous industry references cited at the end of the document, and most notably from NIST AI 100-2 E2023 titled "Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations" [Barrett et al., 2023].

With these definitions and taxonomy, conversations regarding the evaluation of AI threats and risks, developing appropriate control measures, and governing responsible AI development can advance with greater clarity and consistency across diverse CSA groups and among stakeholders. Establishing a common nomenclature reduces confusion, helps connect related concepts, and facilitates more precise dialogue. This document consolidates key terms into a central reference serving the purpose of aligning both the AI Control Working Group and the AI Tech and Risk Working Group within the CSA AI Safety Initiative.

Relationship with the CSA AI Control Framework

The CSA AI Control Framework Working Group's goal is to define a framework of control objectives to support organizations in their secure and responsible development, management, and use of AI technologies. The framework will assist in evaluating risks and defining controls related to Generative AI (GenAI), particularly LLMs.

The control objectives will cover aspects related to cybersecurity. Additionally, it will cover aspects related to safety, privacy, transparency, accountability, and explainability as far as they relate to cybersecurity. Please review CSA's blog post to explore the differences and commonalities between AI Safety and AI Security.

By focusing on the business-to-business implications, the CSA AI Control Framework complements government efforts[1] in protecting national security, citizen's rights and legal enforcement, advocating for secure and ethical AI applications that comply with global standards and regulations.

[1] E.g. EU AI Act, U.S. Artificial Intelligence Safety Institute (USAISI), etc.

1. Large Language Model Assets

This section defines the foundational components essential for implementing and managing LLM systems, from the detailed data assets crucial for training and fine-tuning these models, to the complex LLM-Ops environment, ensuring seamless deployment and operation of AI systems. Furthermore, this section clarifies the LLM's significance, architecture, capabilities, and optimization techniques (see Figure 2).

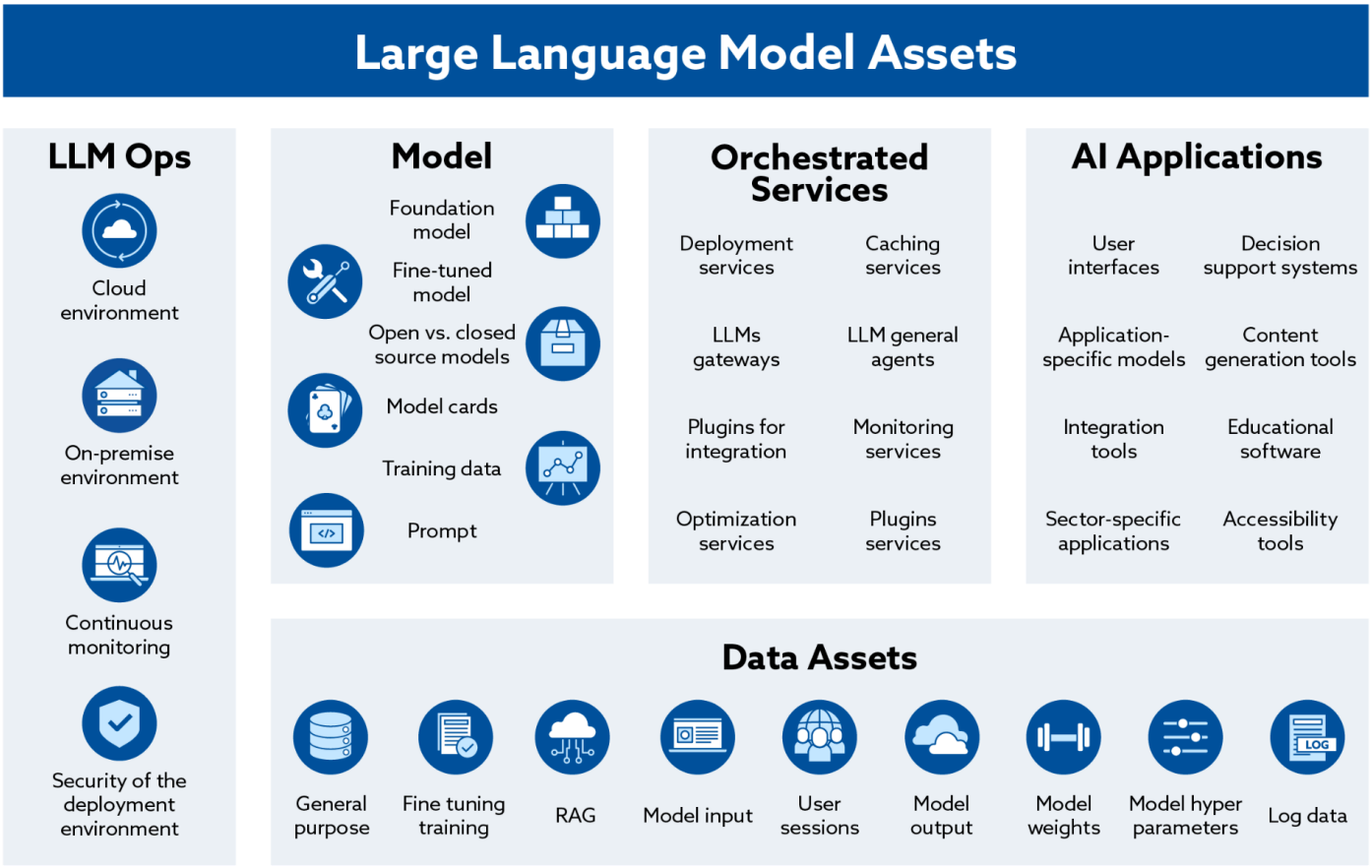

Figure 2: LLM Assets

1.1. Data Assets

In LLM services, many assets play an integral role in shaping a service's efficacy and functionality. Data assets are at the forefront of these assets and serve as the cornerstone of LLM operations. The list below describes the typical range of assets constituting an LLM Service:

- Data used for training, benchmarking, testing, and validation
- Data used for fine-tune training

- Data used for Retrieval-Augmented Generation (RAG)
- Data cards that define the metadata of the data in use
- Input data
- User session data
- Model output data
- Model parameters (weights)
- Model hyperparameters
- Log data from LLM systems

The following are the definitions of these assets:

1. Training, benchmarking, testing, and validation data: This encompasses the data set used to train, benchmark, test, and validate the model, consisting of text sources from which the model derives insights into language patterns, and semantics that are imperative for quality of the model. Each data element is treated and managed individually.

2. Fine-tune training data: Additional data is employed to fine-tune or further pre-train the model post-initial training. This facilitates adjustments to the model's parameters to align more closely with specific use cases or domains, enhancing its adaptability and accuracy.

3. Retrieval-Augmented Generation (RAG): Integrates external knowledge bases with LLMs. By retrieving relevant information before generating responses, RAG enables LLMs to leverage both model knowledge and external knowledge effectively. RAG can retrieve supplementary data from various sources, including internal systems, and public sources, such as the Internet, enriching input prompts and refining the model's contextual understanding to produce higher-quality responses.

4. Data cards: Metadata of the data sets used for various purposes in LLM needs to be maintained. This helps govern AI data and provides lineage, traceability, ownership, data sensitivity, and compliance regimes for every data set used. Storing and then continuously updating data cards as the data, ownership, or requirements change is essential to maintain compliance and visibility.

5. Input data (system-level prompt): The input data is provided to set the context and boundaries around LLM systems. These data sets are additionally used to set topic boundaries and guardrails in case of adversarial generation.

6. User session data: Information amassed during user interactions with the AI systems, encompassing input queries, model-generated responses, and any supplementary context provided by users, facilitating personalized interactions.

7. Model output data: The resultant output generated by the model in response to input prompts, encompassing text responses, predictions, or other forms of processed data, reflective of the model's comprehension and inference capabilities.

8. Model parameters (weights): Internal parameters or weights acquired by the model during training, delineating its behavior and exerting a profound influence on its capacity to generate and contextually relevant responses.

9. Model hyperparameters: Configurations or settings specified during model training, including parameters such as learning rate, batch size, or architecture choices, are pivotal in shaping the model's overall performance and behavior.

10. Log data: Recorded data encapsulating various events and interactions during the model's operation, including input prompts, model responses, performance metrics, and any encountered errors or anomalies, instrumental for monitoring and refining the model's functionality and performance.

1.2. LLM-Ops Cloud Environment

The LLM-Ops Environment encompasses the infrastructure and processes involved in the deployment and operation of LLMs. The following bullet points are the key terms associated with this environment:

- Cloud running the training environment
- Cloud running the model inference point
- Cloud running the AI applications
- Hybrid and multi-cloud infrastructure
- Security of the deployment environment
- Continuous monitoring
- Cloud to host training data (Storage)

The significance and essence of each of the above asset within the framework of the LLM-Ops Environment is described below:

1. Cloud running the training environment: This denotes the cloud platform or service provider entrusted with hosting and managing the computational resources, storage facilities, and ancillary infrastructure pivotal for training LLMs. It serves as the development space where models undergo iterative refinement and enhancement.

2. Cloud running the model inference point: This encapsulates the cloud platform or service provider tasked with hosting and administering the computational resources, storage solutions, and associated infrastructure indispensable for deploying LLMs and facilitating inference processes. It enables the model to generate responses based on user inputs, ensuring seamless interaction and responsiveness.

3. Public/Private/Hybrid Cloud Running the AI applications: This refers to the cloud platform or service provider entrusted with hosting and overseeing the infrastructure essential for running AI applications or AI services, harnessing the capabilities of trained language models. It serves as the operational hub where AI-driven applications leverage the inference prowess of models to deliver value-added functionalities and services to end-users.

4. Security of the deployment environment: This encompasses the array of mechanisms and policies implemented to govern and fortify access to the assorted components of the LLM-Ops Environment. It encompasses Identity and Access Management (IAM) protocols and network security measures, safeguarding the integrity and confidentiality of critical assets and functionalities.

5. Continuous monitoring: This denotes the ongoing process of vigilantly scrutinizing the LLM-Ops Environment's performance, security posture, and overall well-being. It encompasses the vigilant surveillance of the training environment, inference end point, and application components, ensuring optimal functionality while promptly identifying and remedying any anomalies or issues that may arise.

6. Cloud to host training data (Storage): This signifies the cloud platform or service provider tasked with securely housing and managing the extensive data sets requisite for training language models. It entails robust storage and data management capabilities to accommodate the voluminous and diverse data sets fundamental for nurturing and refining language models.

1.3. Model

The concept of "Model" in the context of ML refers to a mathematical representation or an algorithm trained to make predictions or perform a specific task.

The choice of foundation model, fine-tuning approach, and the decision to use open-source or closed-source models can significantly affect LLMs' capabilities, performance, and deployment flexibility within various applications and domains.

We define the following model assets in this subsection:

- Foundation Model
- Fine-Tuned Model
- Open Source vs. Closed Source Models
- Domain-Specific Models
- Model cards

1. Foundation Model:
The Foundation Model is the base upon which further advancements are built. These models are typically large, pre-trained language models that encapsulate a broad understanding of language, obtained from extensive exposure to unlabeled text data through self-supervised learning techniques. Foundation models, in general, provide a starting point for subsequent fine-tuning and specialization to cater to specific tasks or domains. For some advanced and innovative foundation models, another term, "Frontier Model" can be used to represent a brand new foundation model in the AI Marketplace. From an AI perspective, sometimes the term "Base Model'' represents foundation models in the application technology stacks.

2. Fine-Tuned Model:
Derived from the Foundation Model, the Fine-Tuned Model undergoes refinement and adaptation to cater to specific tasks or domains. Through the process of fine-tuning, the parameters of the foundation model are updated utilizing supervised learning techniques and task-specific labeled data. This iterative process enables the model to enhance its performance on target tasks or domains while retaining the foundational knowledge and capabilities inherited from the Foundation Model.
3. Open-Source vs. Closed-Source Models:
This dichotomy pertains to the accessibility and licensing of a model's source code, model weights, and associated artifacts. Open-source models may release some or all of their training data and source code,

data used for the model development, model architecture, weights, and tools to the public under open-source licenses, granting free usage with specific terms and conditions. However, closed-source models maintain proprietary status, withholding their source code, weights, and implementation details from the public domain, often motivated by intellectual property protection or commercial interests. Closed-source models that allow users to access the models for fine tuning or inference purposes are called Open access models.

These model assets collectively form the backbone of model development, fostering innovation, adaptability, and accessibility within GenAI.

4. Domain-Specific Models:
Domain-specific models refer to machine learning models that are designed and trained to excel on specific domain knowledge, such as financial, medicines, and coding.

5. Model cards:
The characteristics of models can be described using model cards. Model cards are files that maintain the context of the model which is essential for Governance and making sure AI models can be used correctly. Model cards[2] consist of model context details like ownership, performance characteristics, data sets the model is trained on, order of training etc. This also helps with traceability, lineage and understanding the behavior of the model. Model cards need to be continuously maintained and updated as the context metadata changes. [CSA, 2024]

More details of model cards can be found, for example, at the Hugging Face platform, where the machine learning community collaborates on models, datasets, and applications.

1.4. Orchestrated Services

These services encompass a range of components and functionalities that enable the efficient and secure operation of LLMs.

The following is the list of Orchestrated Services Assets:

- Caching Services
- Security Gateways (LLM Gateways)
- Deployment Services
- Monitoring Services
- Optimization Services
- Plug-ins for Security
- Plug-ins for Customization and Integration
- LLM General Agents

Definition and significance of each of the above listed assets within the context of orchestrated services follows below.

[2] For more details on 'Model cards' please consult the 'AI Model Risk Management Framework' of the AI Risk and Technology working group.

1. Caching Services:
Caching Services refer to systems or components that facilitate the caching of model predictions, inputs, or other data to enhance performance by reducing redundant computations. By temporarily storing frequently accessed data, caching services help minimize response times and alleviate computational strain on LLMs.

2. Security Gateways (LLM Gateways):
Security Gateways, also known as LLM Gateways, are specialized components that serve as intermediaries between LLMs and external systems. These gateways bolster security by implementing access control measures, input validation, filtering malicious content (such as prompt injections), PII/privacy information, and safeguards against potential threats or misuse, ensuring the integrity and confidentiality of data processed by LLMs.

3. Deployment Services:
Deployment Services streamline the deployment and scaling of LLMs across diverse environments, including cloud platforms and on-premises infrastructure. These services automate deployment processes, facilitate version management, and optimize resource allocation to ensure efficient and seamless LLM deployment.

4. Monitoring Services:
Monitoring Services are pivotal in overseeing LLM security, performance, health, and usage. These services employ monitoring tools and techniques to gather real-time insights, detect anomalies, misuse (such as prompt injections) and issue alerts, enabling security, proactive maintenance, and timely intervention to uphold the optimal operation of LLMs.

5. Optimization Services:
Optimization Services are geared towards optimizing the performance and resource utilization of LLMs. These services employ a range of techniques such as model quantization, pruning, efficient inference strategies to enhance LLM efficiency, reduction of computational overhead, and improvement of overall performance across diverse deployment scenarios.

6. Plug-ins for Security:
Security plug-ins extend LLM security by providing data encryption, access control mechanisms, threat detection capabilities, and compliance enforcement measures, thus increasing cyber resiliency.

7. Plug-ins for Customization and Integration:
Plug-ins for Customization and Integration enable the customization of LLM behavior and seamless integration with other systems, applications, or data sources. These plug-ins provide flexibility in tailoring LLM functionalities to specific use cases or domains and facilitate interoperability with existing infrastructure, fostering enhanced versatility and utility of LLM deployments.

8. LLM General Agents:
LLM General Agents are intelligent agents or components collaborating with LLMs to augment their functionalities and capabilities. These agents may perform various tasks, such as

- planning,
- reflection,

- function calling,
- monitoring,
- data processing,
- explainability,
- optimization,
- scaling, and collaboration,
- and enhancing the versatility and adaptability of LLM deployments in diverse operational contexts.

1.5. AI Applications

AI applications have become ubiquitous, permeating various facets of our daily lives and business operations. From content generation to language translation and beyond, AI applications fueled by LLMs have revolutionized industries and reshaped how we interact with information and technology. However, with the proliferation of AI applications comes the imperative need for effective control frameworks to govern their development, deployment, and usage.

AI applications represent the pinnacle of innovation, offering many capabilities that cater to diverse business domains and use cases. These applications leverage the power of LLMs to decipher and process natural language inputs, enabling functionalities such as content generation, question answering, sentiment analysis, language translation, and more. Essentially, AI applications serve as the interface through which users interact with the underlying intelligence of LLMs, facilitating seamless communication and task automation across various domains.

As downstream applications of LLMs, AI applications are one of the most important assets to consider in an AI control framework. They represent the direct touchpoint between LLM technology and end-users, shaping how users perceive and interact with AI systems. As such, AI applications have the potential to amplify the benefits or risks associated with LLMs.

AI applications can have significant economic impacts. As businesses increasingly rely on AI applications to drive innovation, streamline operations, and gain competitive advantages, the responsible development and deployment of these applications become crucial for maintaining market integrity and fostering a level playing field.

Given these considerations, an AI control framework must prioritize the governance and oversight of AI applications. This includes establishing guidelines and standards for AI application development, testing, deployment, operation, and maintenance, ensuring compliance with relevant regulations, and promoting transparency and accountability throughout the AI application lifecycle. Additionally, the framework should facilitate continuous monitoring and evaluation of AI applications, enabling timely identification and mitigation of potential risks or unintended consequences.
By prioritizing AI applications in the AI control framework, organizations can proactively address the challenges and risks associated with LLM-powered applications while unlocking their transformative potential to drive innovation and improve lives.

AI application cards are files that maintain the AI context of the application which is essential for governance of the application. AI application cards convey the AI data of the applications, including models used, data sets used, application and AI use cases, application owners, and guardians. AI application cards are an easy way to convey and share AI data for applications, to help AI governance executives, AI councils, and regulators to understand the application and the AI it uses. The AI application cards may in turn point to model and data cards.

2. LLM-Service Lifecycle

The LLM-Service Lifecycle outlines distinct phases, each crucial in ensuring the service's efficiency, reliability, and relevance throughout its lifespan. From the preparatory stages of conceptualization and planning to the eventual archiving and disposal, each phase is intricately integrated into a comprehensive framework designed to improve service delivery and maintain alignment with evolving needs and standards. Organizations can manage service development, evaluation, deployment, delivery, and retirement through this structured approach with clarity and effectiveness.

Drawing upon emerging standards like ISO/IEC 5338 on AI system lifecycles, and reviews from organizations like the UK's Centre for Data Ethics and Innovation (CDEI), this lifecycle covers the end-to-end process, from early preparation and design through training, evaluation, deployment, operation, and eventually retirement.

The following is the high-level breakdown of the lifecycle we will define in this section.

- **Preparation:**
  - Data collection
  - Data curation
  - Data storage
  - Resource provisioning
  - Team and expertise

- **Development:**
  - Design
  - Training
  - Key considerations during development
  - Guardrails

- **Evaluation/Validation:**
  - Evaluation
  - Validation/Red Teaming
  - Re-evaluation
  - Key considerations during evaluation/validation

- **Deployment:**
  - Orchestration
  - AI Services supply chain
  - AI applications
- **Delivery:**
  - Operations
  - Maintenance
  - Continuous monitoring
  - Continuous improvement

- **Service Retirement:**
    - Archiving
    - Data deletion
    - Model disposal

2.1 Preparation

This phase lays the foundation for the entire LLM development process and greatly influences the model's quality and ethical behavior. It begins with careful data considerations.
In this subsection, we define the following terms:

- Data collection
- Data curation
- Data storage
- Resource provisioning
- Team and expertise

Data collection should focus on identifying diverse, sufficiently large, high-quality data sources (text, code, etc.) that align with the LLM's purpose. Ethical sourcing practices and potential biases in the data must be considered. It's important to determine the data needed for effective training and ensure it reflects the real-world contexts where the LLM will be deployed to avoid biased or discriminatory outputs.

Data curation is the process that enhances data quality. This includes cleaning (removing errors, inconsistencies, and irrelevant information), categorization (organizing data by logical topics or themes), classification (assigning labels for supervised learning), labeling, anonymization, and transformation (changing data formats for compatibility).

Data storage must ensure accessibility through solutions like databases or cloud storage services while prioritizing strong security measures to protect sensitive data and comply with privacy regulations.

Resource Provisioning in the preparation phase involves selecting appropriate compute and cloud resources. Hardware choices should consider processors (CPUs, GPUs, TPUs) and memory-optimized for the LLM, while software selections include reliable operating systems, libraries, and programming languages suitable for LLM development. Cloud infrastructure might be leveraged for scalability, flexibility, and cost efficiency.

Team and Expertise is crucial. Data scientists gather, process, and analyze data, machine learning engineers design and fine-tune the LLM, software developers create necessary tools, linguists offer language expertise, and ethicists evaluate the model's social impacts and ways to mitigate risks.

Key considerations during preparation should always include clearly defining the LLM's purpose to guide responsible choices, proactively identifying and addressing potential biases in the data, and implementing strong privacy measures throughout the data lifecycle. A data chain-of-custody needs to be a "must have" for secure data work and model development. Steps to ensure the training data is not tampered with at each phase of collection, curation, and storage cannot be ignored.

2.2 Development

This phase transforms the prepared data and computational resources into a functional LLM.

Key activities include:

- Design
- Development supply chain
- Training
- Key considerations during development

2.2.1 Design

Model Architecture: Choose a suitable LLM architecture (e.g., Transformer-based, Recurrent Neural Networks) based on the model's intended tasks. Consider factors like performance requirements, computational constraints, and the type of data the model will process.

Hyperparameter Selection: Determine optimal hyperparameters (learning rate, batch size, number of layers, etc.) that govern the model's training process. These choices impact training time, convergence, and the model's accuracy.

Evaluation Metrics: Define metrics (e.g., accuracy, perplexity, BLEU score) to track the model's performance during training and identify areas for improvement.

2.2.2 Development Supply Chain

Foundation Models: Consider leveraging pre-trained foundation models (e.g., GPT-3, BERT) that offer a strong starting point and fine-tune them with your specific data for tailored results.

Components: Evaluate the need for specialized components for tasks like named entity recognition, sentiment analysis, or text summarization. Choose pre-built open-source or closed-source components or develop custom ones.

Frameworks: Select a machine learning framework (e.g., TensorFlow, PyTorch, Ray) that simplifies model development, training, and deployment.

2.2.3 Training

Training process: Feed the curated data into the chosen model architecture. Update the model's parameters iteratively, using an optimization algorithm (like gradient descent) to minimize errors in the training data.

Monitoring: Closely monitor training progress with the defined evaluation metrics. Look for signs of overfitting or underfitting and adjust the training strategy or hyperparameters accordingly.

Experimentation: Adopt an iterative approach. Test different model architectures, hyperparameters, and data preprocessing techniques to find the best configuration.

Tokenization: This involves breaking down the input text into smaller units called tokens, which can be individual words, subword units, or even individual characters. Its main purpose is to convert raw text into a numerical format that can be processed by the LLM's neural network by mapping each token to a unique integer value or embedding vector. Tokenization directly affects how the input text is represented and processed by the model, making it a fundamental step in the pipeline of LLMs, as proper tokenization can improve the model's ability to understand and generate natural language while ensuring computational efficiency.

2.2.4 Key Considerations During Development

Transparency: Document design choices, model architectures, and training procedures to support reproducibility and accountability.

Explainability: Use techniques that help interpret the model's outputs where possible. This is especially important in high-stakes applications.

Efficiency: Balance the LLM's performance with computational resource consumption. Explore optimization techniques (like quantization or pruning) to improve efficiency without sacrificing accuracy.

Version Control: Implement robust version control practices to track changes to the model, tokenization strategies, training datasets, and other components. This helps ensure reproducibility, enables rolling back to previous versions if needed, and facilitates collaboration among developers.

2.3 Evaluation/Validation

This phase rigorously assesses the LLM's performance, reliability, and suitability for its intended purpose before deployment.

We define the following terms:

- Evaluation
- Validation/Red Teaming
- Re-evaluation
- Key considerations during evaluation/validation

2.3.1 Evaluation

Metrics: Use a combination of quantitative and qualitative metrics tailored to the LLM's task. Quantitative metrics include accuracy, precision, recall, F1-score, perplexity (for language generation), and BLEU score (for translation). Qualitative metrics may involve human judgment of outputs for fluency, coherence, and relevance.

Benchmarking: Compare the LLM's performance to established baselines or other state-of-the-art models to understand its relative strengths and weaknesses.

Bias and Fairness Testing: Examine the model's output for potential biases across various demographic groups or sensitive attributes. Use fairness metrics to quantify disparities.

2.3.2 Validation/Red Teaming

Real-world Testing: Test the LLM in realistic scenarios resembling its intended use case. Evaluate its performance on unseen data to assess generalization capabilities.

Human-in-the-loop: Involve human experts to evaluate the LLM's outputs, especially in sensitive domains where accuracy and nuance are critical. Collect feedback to guide future refinements.

Red Teaming: Employ an adversarial team to intentionally probe the LLM's vulnerabilities, biases, and failure modes. This approach can uncover weaknesses that may be missed during regular testing.

2.3.3 Re-Evaluation

Monitoring: Continuously monitor the LLM's performance after deployment. Implement mechanisms to detect potential data and model drift or declining performance over time.

Data Drift refers to a situation where the distribution of the input data changes over time, causing the model's performance to degrade. This can happen when the real-world data evolves in a way that deviates from the data used to train the model. As a result, the model's predictions become less accurate and reliable.

Model Drift occurs when the relationship between the input features and the target variable changes over time, causing the model to become less effective at making predictions. This can happen due to various reasons, such as changes in the underlying process that generated the data, shifts in consumer behavior, or external factors like economic conditions or regulations.

Both data and model drift can lead to a degradation in the performance of machine learning models, and it's essential to monitor for these issues and take appropriate measures to mitigate them. Techniques like continuous monitoring, retraining, or updating the model with new data can help address data and model drift.

Triggering Re-training: Establish criteria for when performance degradation or shifts in the data distribution warrant a full or partial re-training of the LLM.

2.3.4 Key Considerations During Evaluation/Validation

Resilience: Evaluate the LLM's capacity to withstand adversarial inputs, outliers, and anomalous data patterns, ensuring consistent performance even in unforeseen circumstances.

Uncertainty: Explore methods to express the model's confidence in its predictions. This can guide human decision-making when using the LLM in real-world tasks.

Data representativeness: Use evaluation data sets that closely mirror the real-world data the LLM will encounter in operation. Failure to do so can give a misleading picture of performance.

2.4 Deployment

This phase involves integrating the trained and validated LLM into operational systems where it can provide its intended service.

This subsection defines the following terms:

- Orchestration
- AI Services supply chain
- Applications
- Key considerations during deployment
- Guardrails

2.4.1 Orchestration

Containerization: Package the LLM and its dependencies (libraries, data, etc.) into a self-contained unit (e.g., a Docker container) for portability and simplified deployment across different environments.

Scalability: Design a deployment architecture that can scale up or down based on demand. Consider load-balancing techniques to distribute incoming requests efficiently.

Versioning: Implement a system to track different LLM versions, their configurations, and performance metrics. This will facilitate rollbacks and comparisons when deploying updates.

IaC: Use infrastructure as code for orchestration of infrastructure. This gives many advantages like traceability of changes, easier rollbacks, and so on.

2.4.2 AI Services Supply Chain

Agents: If the LLM is part of a larger conversational AI system, determine how it will interact with other components like natural language understanding (NLU) modules, dialogue managers, and knowledge bases.

Plug-ins: Integrate the LLM with necessary plugins or extensions to enhance its functionality (e.g., plugins for specific domains like healthcare or finance). Consider the security implications of integrating external components.

Security: Prioritize security measures throughout the supply chain. Protect API endpoints, implement user authentication/authorization protocols, tokenize access credentials, and encrypt sensitive data in transit and at rest.

2.4.3 Applications

Application Programming Interfaces (APIs): Develop well-structured APIs to allow external entities (systems and users) to interact with the LLM-powered applications. Provide clear documentation of input/output formats and expected behavior. APIs should be built using standards like REpresentation State Transfer (REST) and have version controls.

Retrieval-Augmented Generation (RAG): If applicable, consider integrating a retrieval component to enable the LLM to access and incorporate relevant information from external knowledge sources for more informed responses.

Prompt Injection: Explore techniques for injecting prompts or instructions to guide the LLM's behavior towards specific tasks or to constrain outputs for safety.

Insecure Output Handling: Double-checking the output is essential to avoid harmful outputs that could lead to downstream security exploits, including code execution that compromises systems and exposes data.

2.4.4 Key Considerations During Deployment

User Interface/User Experience (UI/UX): Design user-friendly interfaces to interact with LLM-powered applications. Consider the context where the LLM will be used and tailor the interface accordingly.

Observability: Establish logging and monitoring systems to track API usage, LLM performance, and error rates. This data will be helpful for debugging and optimization.

Transparency: Provide users with some level of visibility into how the LLM works and the potential limitations of its outputs. This builds trust and understanding.

Input Filters: Identify and prevent malicious prompts from being sent into the LLM, to reduce toxic content output.

Output Filters: Prevent the generation of inappropriate or harmful content, including hate speech, violence, explicit material, and other types of content that are considered unacceptable or harmful.

Privacy: Provide users with controls to mitigate privacy risks, such as PII not improperly used or disclosed. This includes measures to prevent the models from generating responses that might reveal personal information about individuals or proprietary information.

Misuse: Limitations on the usage of LLMs to prevent misuse, such as generating deceptive content, phishing emails, or other forms of manipulative or unethical content.

Ethical Guidelines and Bias Mitigation: Ensure LLMs are used in a manner that is consistent with ethical principles and societal norms. Reduce the likelihood of generating biased or discriminatory content. This includes biases related to race, gender, sexuality, and other personal characteristics.

2.5 Delivery

This phase focuses on the ongoing management of the deployed LLM and iterative improvements to maintain its value and performance.

This includes the following three sub-phases that we define in this section:

- Operation
- Maintenance
- Continuous improvement

2.5.1 Operation

Logging and Monitoring: Continuously track the LLM's performance using established metrics for accuracy, latency, and resource utilization and implement alert systems to notify relevant personnel of any security issues or performance degradation.

Incident Response: Establish plans and procedures to promptly address and resolve system failures, and security incidents, such as cyber-attacks, bugs or performance bottlenecks.

User Feedback: Establish mechanisms to collect user feedback about the LLM's outputs. Analyze this feedback to identify areas for improvement or potential issues.

2.5.2 Maintenance

Bug Fixes: Address errors or malfunctions within the code which are used to train, fine tune, and deploy LLM models or its supporting systems. Release patches or updates to ensure the system's stability and integrity.

Security Updates: Stay vigilant against emerging security threats and vulnerabilities. Patching process should include inventory of third-party/public LLMs being used, and their versions. Patch (i.e., provide security updates for) LLMs and related systems based on the established vulnerability management SLAs.

Retrain Model: As the nature of the data the LLM interacts with changes, you may need to update the training data or retrain the model to maintain optimal performance.

2.5.3 Continuous Improvement

Re-training: Regularly evaluate the need to retrain the LLM on new data or with updated hyperparameters. This can help address concept drift, and performance declines, or expand the LLM's capabilities.

Continuous Feedback Loop: Implement a feedback loop where insights from monitoring and user feedback guide re-training and refinement efforts.

Experimentation: Continuously explore new model architectures, algorithms, or training techniques that could potentially improve the LLM's overall performance.

2.5.4 Key Considerations During Delivery

Throughout the operation and maintenance process, it's important to actively monitor LLMs for any unintended harmful behaviors or biases that might surface post-deployment. This proactive approach ensures that issues are identified promptly, allowing for necessary adjustments or interventions to mitigate potential negative impacts on users or systems.

Change management is pivotal in maintaining the LLM's stability and performance. Maintaining a comprehensive change management process is essential, as well as documenting all updates and tracking their performance impact. By having robust change management procedures in place, organizations can effectively manage the evolution of the LLM while minimizing disruptions. Additionally, readily available rollback plans enable swift action in the event of unexpected issues arising from updates or modifications.

Planning for potential downtime scenarios is another critical aspect of LLM maintenance. Organizations should anticipate the need for updates or maintenance that may require temporary service interruptions. Proper communication of downtime windows with the LLM's users is essential to minimize disruptions and ensure that stakeholders are informed and prepared for any potential impact on their operations.

By proactively addressing downtime considerations, organizations can maintain the reliability and availability of the LLM while meeting user expectations and requirements.

2.6 Service Retirement

This phase focuses on properly decommissioning the LLM service when it's no longer needed, superseded by a newer model, or if its continued operation poses unacceptable risks.

The following terms are defined below in the section:

- Archiving
- Data deletion
- Model disposal

2.6.1 Archiving

Model Preservation: Archive the LLM model itself, along with its relevant code, configuration files, and training data. This can be valuable for historical analysis, auditing, or potential future reuse. This archive should be stored according to organizational data retention policies.

Documentation: Preserve thorough documentation of the LLM's design, development process, performance metrics, known limitations, and any incidents or ethical concerns encountered during its use.

2.6.2 Data Deletion

Regulations: Comply with data governance regulations (e.g., GDPR, CCPA) regarding the secure deletion of any personal or sensitive data collected or used in training or prompting during the LLM's operation.

Retention Policies: Implement clear data retention policies that determine how long data needs to be stored and when it should be securely disposed of.

2.6.3 Model Disposal

Reuse Assessment: Determine if the LLM or its components can be repurposed for other applications or research projects, potentially reducing development costs and environmental impact.

Intellectual Property: Address any intellectual property considerations related to retiring a model, particularly if it was developed using external resources or licensed technology.

Secure Disposal: If the LLM cannot be reused, it should be securely disposed of, preventing unauthorized access or potential misuse. If the LLM was stored on physical media (e.g., hard drives, SSDs,

or removable storage), consider physically destroying the media to ensure that no data can be recovered. This can be done through degaussing, shredding, or physically destroying the media beyond recovery.

2.6.4 Key Considerations During Service Retirement

Communication: Notify users and stakeholders before an LLM service retires. Provide clear instructions for migrating to alternative services or solutions if needed.

Ethics of Decommissioning: Carefully evaluate the potential impacts of retiring an LLM service, especially for services in sensitive domains or where users have come to rely on it.

Knowledge Transfer: Ensure expertise and lessons learned from the retired LLM's development and operation are effectively passed on to inform future AI projects within the organization.

3. LLM-Service Impact Categories

We can map the impact categories directly onto the well-established CIA triad (Confidentiality, Integrity, and Availability). This could be expanded to include the new categories of 'Abuse/Misuse' and 'Loss of Privacy" (according to the NIST document AI 100-2 E2023).
What follows is the initial list of high-level impact categories of LLM-related risks:

- **Loss of Confidentiality:** There is a risk that sensitive information in the LLM's data, the model itself, or the output it generates, might be exposed or leaked to unauthorized individuals. This could involve personal data, trade secrets, or other confidential material.
- **Loss of Integrity:** The risk that the LLM's data or its generated outputs are altered or corrupted. This could be done maliciously or accidentally, leading to incorrect or misleading results.
- **Loss of Availability:** The risk of disruption to the LLM's operation, preventing users from accessing it when needed. Disruption could range from denial-of-service attacks to system failures, unexpected downtime, excessive billing quotas or computational resources.

4. LLM Service Threat Categories

The initial list of LLM Service Threat Categories encompasses a range of potential risks and vulnerabilities that need careful consideration and mitigation strategies. Each category represents a distinct challenge that could compromise the integrity, security, and effectiveness of LLM services. The following lists each category:

1. Model manipulation
2. Data poisoning
3. Sensitive data disclosure
4. Model theft
5. Model Failure/malfunctioning
6. Insecure supply chain
7. Insecure apps/plugins
8. Denial of Service (DoS)
9. Loss of governance/compliance

4.1. Model Manipulation

This category involves attempts to evade detection or manipulate the LLM model to produce inaccurate or misleading results. It encompasses techniques, such as direct or indirect prompt injection (adversarial inputs), which aim to exploit vulnerabilities in the model's understanding and decision-making processes.

4.2. Data Poisoning

Data poisoning refers to manipulating training data used to train an LLM model. This manipulation can be malicious, with attackers intentionally injecting false, misleading, or unintentional data points, where errors or biases in the original data set are included. In either case, data poisoning can lead to a tainted model that learns incorrect patterns, produces biased predictions, and becomes untrustworthy.

4.3. Sensitive Data Disclosure

This category encompasses threats related to the unauthorized access, exposure, or leakage of sensitive information processed or stored by the LLM service. Sensitive data may include personal information, proprietary data, or confidential documents, the exposure of which could lead to privacy violations or security breaches.

4.4. Model Theft

Model Theft (distillation) involves unauthorized access to, or replication of, the LLM model by malicious actors. Attackers may attempt to reverse-engineer the model architecture or extract proprietary algorithms and parameters, leading to intellectual property theft or the creation of unauthorized replicas.

4.5. Model Failure/Malfunctioning

This category covers various types of failures or malfunctions within the LLM service, including software bugs, hardware failures, hallucinations, or operational errors. Such incidents can disrupt service availability, degrade performance, or compromise the accuracy and reliability of the LLM model's outputs.

4.6. Insecure Supply Chain

An insecure supply chain refers to vulnerabilities introduced through third-party components, dependencies, or services integrated into the LLM ecosystem. Vulnerabilities in the supply chain, such as compromised software libraries or hardware components, can be exploited to compromise the overall security and trustworthiness of the LLM service.

4.7. Insecure Apps/Plugins

This category pertains to vulnerabilities introduced in plugins, functional calls, or extensions that interact with the LLM service. Insecure or maliciously designed applications/plugins may introduce security

loopholes, elevate privilege levels, or facilitate unauthorized access to sensitive resources. Insecure plugins pose risks to both the input and output of integrated systems.

4.8. Denial of Service (DoS)

Denial of Service attacks aim to disrupt the availability or functionality of the LLM service by overwhelming it with a high volume of requests or malicious traffic. DoS attacks can render the service inaccessible to legitimate users, causing downtime, service degradation, or loss of trust.

4.9. Loss of Governance/Compliance

This category involves the risk of non-compliance with regulatory requirements, industry standards, or internal governance policies governing the operation and use of the LLM service. Failure to adhere to governance and compliance standards can result in legal liabilities, financial penalties, or reputational damage.

Addressing the risks of these LLM service threat categories requires a comprehensive approach that includes robust security measures, ongoing risk assessments, threat intelligence integration, and proactive mitigation strategies tailored to an LLM deployment's unique characteristics and challenges.

From a security control and risk management perspective, to address these threat categories, we need to distinguish between weaknesses and vulnerabilities associated with LLM systems.

Weaknesses in LLMs can manifest in various forms, including limitations in training data, biases in algorithms, or vulnerabilities in model architectures. For instance, the reliance of LLMs on statistical patterns in training data can lead to weaknesses in handling language nuances or detecting subtle adversarial inputs.

Vulnerabilities in LLMs refer to specific instances where attackers can exploit these weaknesses to compromise the integrity, confidentiality, or availability of the model or its outputs. These could include vulnerabilities in the model's implementation, such as coding errors or misconfigurations, and vulnerabilities in the training data that could be manipulated to inject biases or craft adversarial examples.

From a risk management perspective, identifying and mitigating risks associated with weaknesses and vulnerabilities in LLMs is important for protecting against potential threats and minimizing their impact. This involves assessing the likelihood and potential impact of attacks on the LLM, prioritizing risks based on their severity, and implementing appropriate security controls to mitigate or transfer these risks to an acceptable level. This should be realized considering Red/Blue Teaming activities as part of the security strategy.

By distinguishing between the weaknesses, vulnerabilities, and attacks, AI control frameworks can provide a structured approach to identifying, assessing, and mitigating risks associated with deploying AI systems. This allows organizations to develop effective strategies for protecting against potential threats, enhancing the resilience of their AI systems, and maintaining trust in their operations.

5. References/Sources

1. BARRETT, A.M., NEWMAN, J., NONNECKE, B., HENDRYCKS, D., MURPHY, E.R. and JACKSON, K. (2023). CLTC Center for Long-Term Cybersecurity UC Berkeley AI Risk-Management Standards Profile for General-Purpose AI Systems (GPAIS) and Foundation Models.[online] UC Berkeley Center for Long-Term Cybersecurity, pp.1–94. Available at: https://cltc.berkeley.edu/wp-content/uploads/2023/11/Berkeley-GPAIS-Foundation-Model-Risk-Management-Standards-Profile-v1.0.pdf

2. Huang, K., Wang, Y., Goertzel, B., Li, Y., Wright, S., Ponnapalli, J. (ed.). (2024). Generative AI Security Theories and Practices. Springer. https://link.springer.com/book/9783031542510

3. CSA. (2024). AI Organizational Responsibilities Working Group. AI Organizational Responsibilities - Core Security Responsibilities. [online] Available at: https://cloudsecurityalliance.org/artifacts/ai-organizational-responsibilities-core-security-responsibilities

4. CSA. (2024). AI Technology and Risk Working Group. The AI Model Risk Management Framework. Available at: https://cloudsecurityalliance.org/research/artifacts?term=artificial-intelligence

5. IBM. IBM Watsonx (2024). AI risk atlas. AI risk atlas. [online] Available at: https://dataplatform.cloud.ibm.com/docs/content/wsj/ai-risk-atlas/ai-risk-atlas.html?context=wx&audience=wdp

6. GOV.UK. (n.d.). AI Foundation Models: initial review. [online] Available at: https://www.gov.uk/cma-cases/ai-foundation-models-initial-review.

7. Andreessen Horowitz. Radovanovic, M.B., Rajko (2023). Emerging Architectures for LLM Applications. [online]. Available at: https://a16z.com/2023/06/20/emerging-architectures-for-llm-applications

8. ENISA. (2020). Artificial Intelligence Cybersecurity Challenges. [online] Available at: https://www.enisa.europa.eu/publications/artificial-intelligence-cybersecurity-challenges.

9. Microsoft. MicrosoftLearn. (2022).Failure Modes in Machine Learning - Security documentation. [online] Available at: https://learn.microsoft.com/en-us/security/engineering/failure-modes-in-machine-learning

10. ISO/IEC 22989:2022. Information Technology – Artificial Intelligence – Concepts & Terminology. [online] ISO. Available at: https://www.iso.org/standard/74296.html.

11. ISO/IEC TR 24028:2020. (2022). Information technology — Artificial intelligence — Overview of trustworthiness in artificial intelligence. Available at: https://www.iso.org/obp/ui/#iso:std:iso-iec:tr:24028:ed-1:v1:en

12. ISO/IEC 27090. Cybersecurity – Artificial Intelligence – Guidance for addressing security threats to artificial intelligence systems. Available at: https://www.iso.org/standard/56581.html

13. ISO/IEC 27091. Cybersecurity and Privacy – Artificial Intelligence – Privacy protection – Guidance for organizations to address privacy risks in artificial intelligence (AI) systems and machine learning (ML) models. Available at: https://www.iso.org/standard/56582.html

14. ISO/IEC 42001:2023. Information technology — Artificial intelligence — Management system. Available at: https://www.iso.org/standard/81230.html

15. ISO/IEC DIS 5338. Information technology — Artificial intelligence — AI system life cycle processes. [online] ISO. Available at: https://www.iso.org/standard/81118.html.

16. ISO. Online Browsing Platform (OBP). Terms and Definitions. Available at: https://www.iso.org/obp/ui

17. The MITRE Corporation. MITRE Atlas. Atlas Matrix. Available at: https://atlas.mitre.org/matrices/ATLAS

18. NIST. NIST AI 100-2 E2023. (2024). Vassilev, A., Oprea, A., Fordyce, A. and Anderson, H. Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations. [online] csrc.nist.gov. Available at: https://csrc.nist.gov/pubs/ai/100/2/e2023/final

19. NIST AI RMF 1.0. AI Risk Management Framework. (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0). [online] doi: https://doi.org/10.6028/nist.ai.100-1.

20. NIST. U.S. Artificial Intelligence Safety Institute (USAISI). Available at: https://www.nist.gov/aisi

21. OWASP. Top 10 for LLM Applications and Generative AI. Available at: https://genai.owasp.org

22. Cornell University. (2023). ARXIV. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J. and Wang, H. Retrieval-Augmented Generation for Large Language Models: A Survey. [online] https://doi.org/10.48550/arXiv.2312.10997.

23. Manral, V. LinkedIn. (2023). Shared Responsibilitu Model. Available at: https://www.linkedin.com/posts/vishwasmanral_generativeai-chatgpt-sharedresponsibility-activity-7084313628614537216-DvLj/

24. European Union. (2024). EU Artificial Intelligence Act. Available at: https://artificialintelligenceact.eu