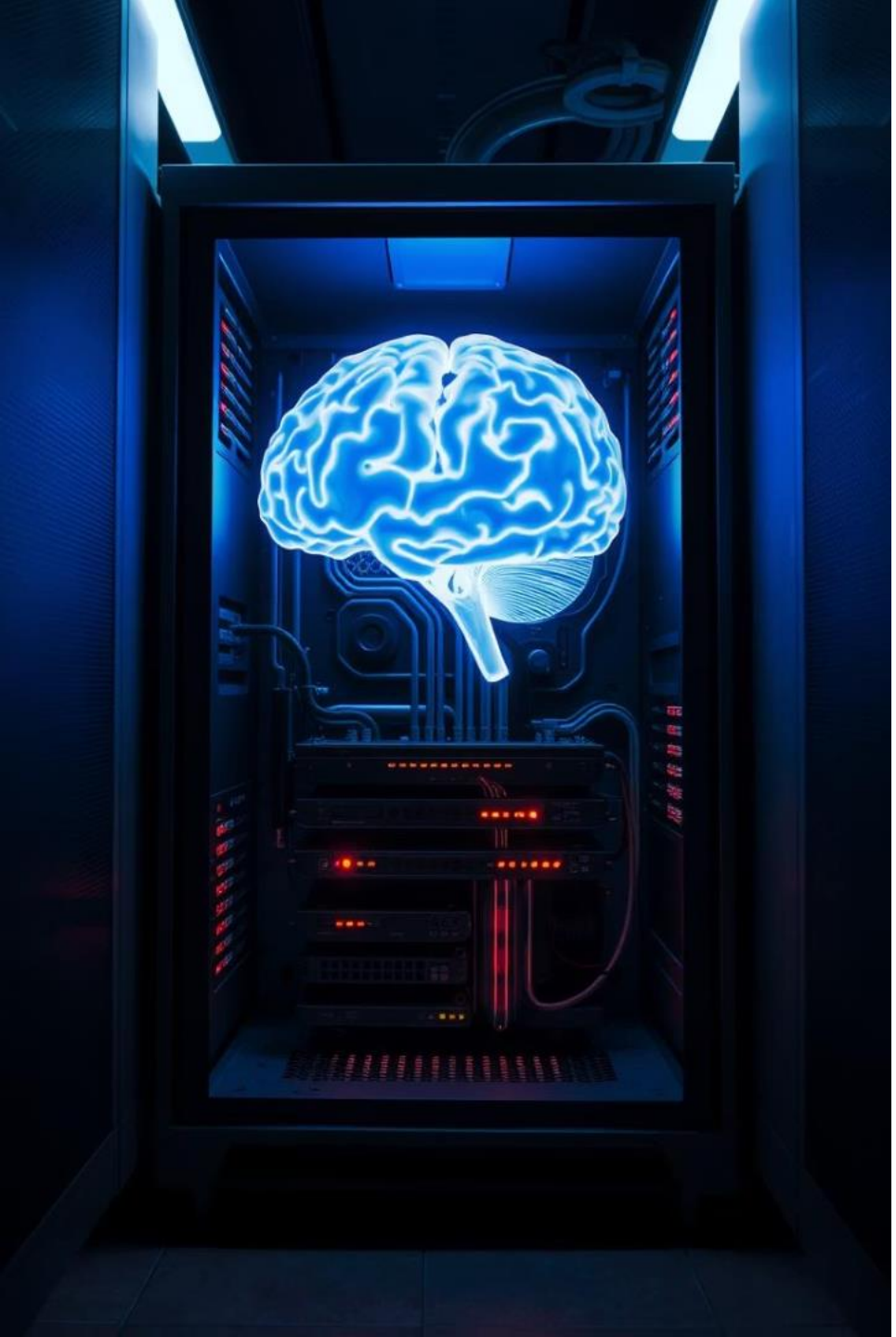


Artificial Intelligence: A Historical Journey and Future Outlook

Artificial Intelligence is rapidly reshaping our world, driving innovations across industries and redefining the boundaries of technology. This presentation will provide a comprehensive overview of AI, tracing its historical roots, exploring key milestones, and examining the current state of the field. We will cover the evolution of AI, from its conceptual beginnings to the cutting-edge developments of today, setting the stage for understanding its future impact.

Harpreet Singh - Enterprise Security Architect

Defining Artificial Intelligence

Core Technologies

At its core, AI is about creating machines that mimic human cognitive functions.

- Visual Perception
- Speech Recognition
- Decision-Making

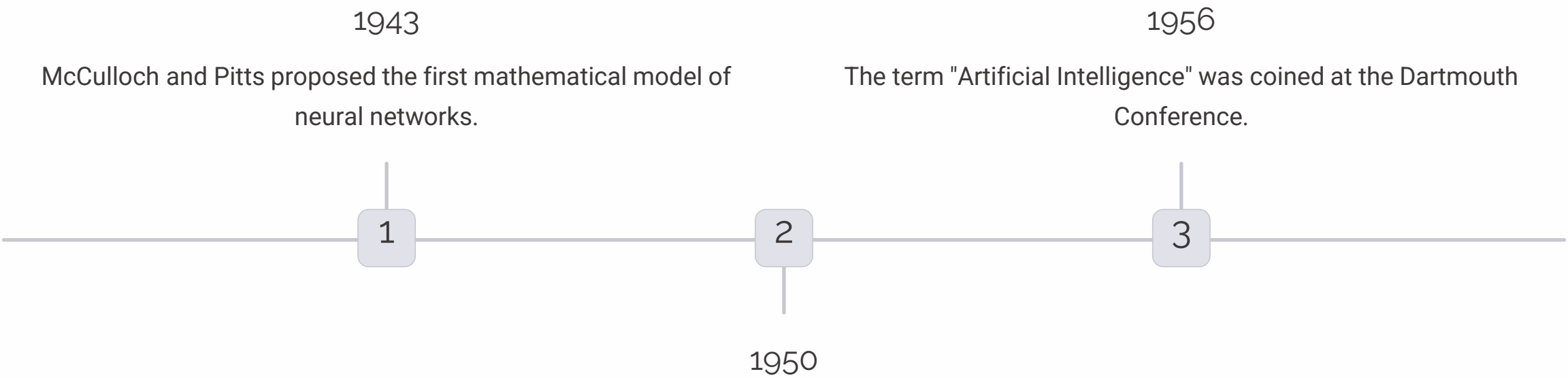
Enabling Capabilities

AI systems are designed to tackle complex tasks.

- Language Translation
- Pattern Recognition
- Problem-Solving

Artificial Intelligence (AI) simulates human intelligence in machines programmed to think and learn like humans. It encompasses a variety of technologies, enabling computer systems to perform tasks that typically require human intelligence.

The Genesis of AI (1940s-1950s)



The 1940s and 1950s marked the theoretical and conceptual birth of AI. Key milestones during this period laid the foundation for future developments. Warren McCulloch and Walter Pitts's neural network model was a critical first step, mathematically simulating brain functions. Alan Turing's work on computability and his proposed Turing Test established a benchmark for machine intelligence, sparking debate and inspiring research. The Dartmouth Conference in 1956, where the term "Artificial Intelligence" was coined, formally launched AI as a field of study.

Early Enthusiasm and Development (1950s-1970s)

1. **Perceptron (1958)**

   Frank Rosenblatt created an early neural network.

2. **Unimate (1961)**

   The first industrial robot was deployed at General Motors.

3. **ELIZA (1964)**

   Joseph Weizenbaum created an early natural language processing program.

4. **DENDRAL (1965)**

   One of the first expert systems was developed at Stanford.

The early years of AI research were filled with optimism. Frank Rosenblatt's Perceptron demonstrated the potential of neural networks. The deployment of Unimate in 1961 marked the first practical application of robotics in industry. ELIZA, Joseph Weizenbaum’s creation, showed the ability of computers to engage in basic natural language interactions. The development of DENDRAL highlighted the potential of expert systems to solve complex problems in specific domains.

AI Winters and Revivals (1970s-1990s)

1

Funding Cuts

Disappointment in AI progress led to reduced interest.

2

Expert Systems

Expert systems became commercially viable in the 1980s.

3

Deep Blue

IBM's Deep Blue defeated Garry Kasparov in 1997.

The journey of AI has been marked by periods of boom and bust. The 1970s and 1980s saw the first "AI Winter," characterized by funding cuts and disillusionment. Despite these challenges, research continued, leading to the rise of expert systems in the 1980s. A significant milestone was achieved in 1997 when IBM's Deep Blue defeated world chess champion Garry Kasparov.

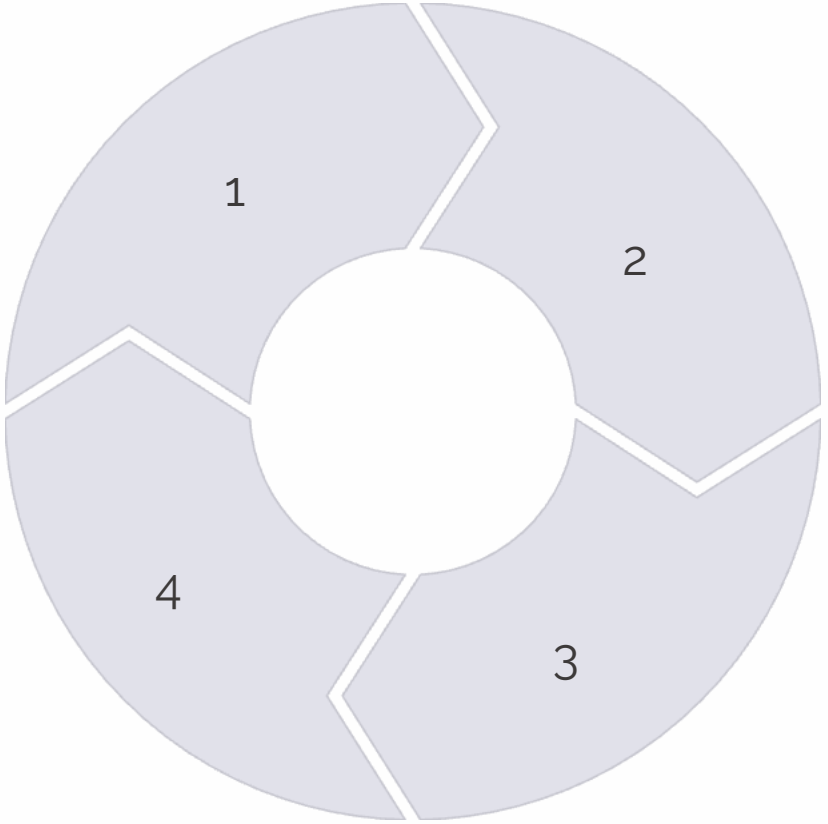
The Big Data Era and Modern AI Revolution (2000s-Present)

Deep Learning (2006)

Geoffrey Hinton introduced deep learning techniques.

IBM Watson (2011)

IBM Watson won Jeopardy! against human champions.



AlphaGo (2014)

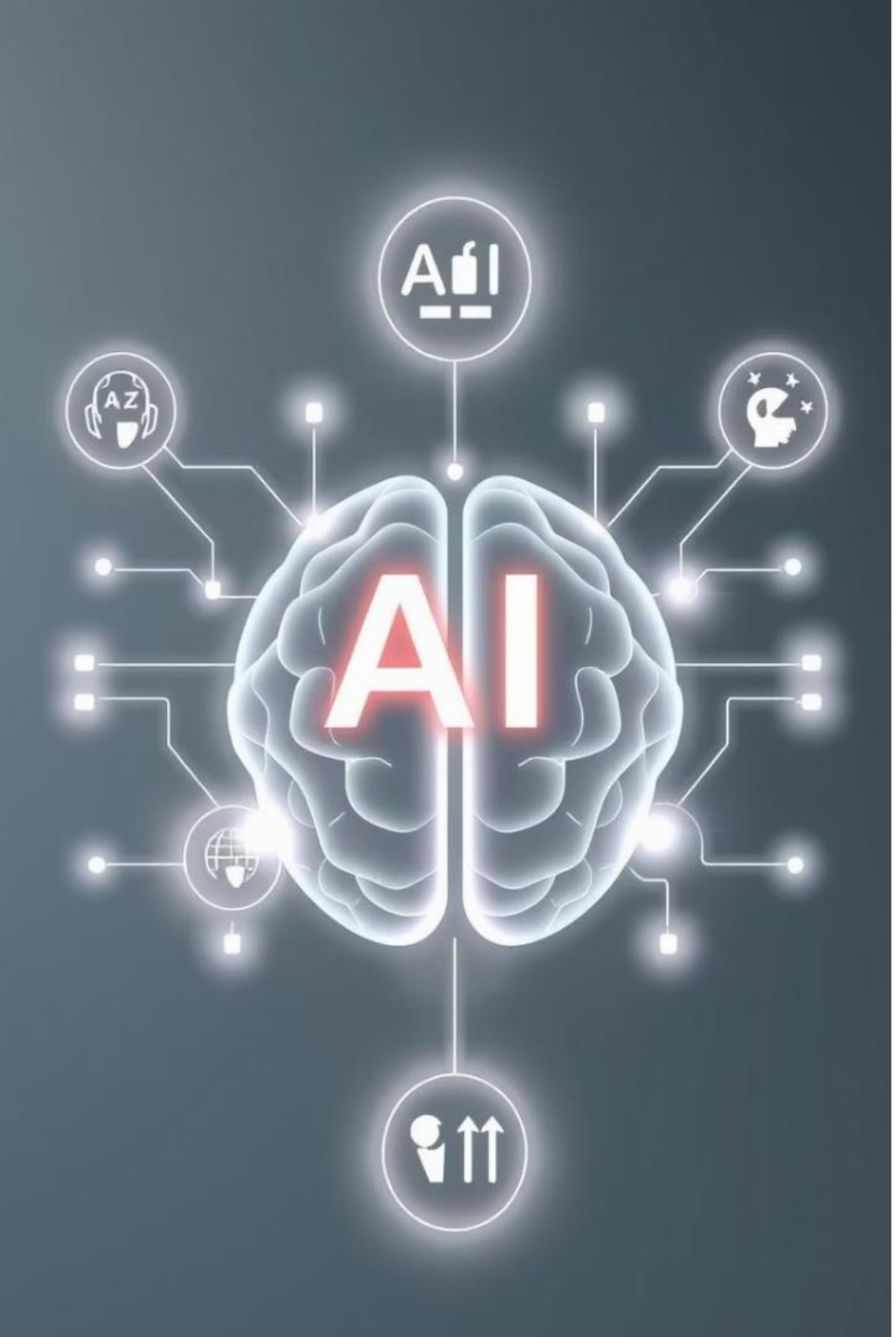
DeepMind's AlphaGo defeated world Go champion Lee Sedol.

AlexNet (2012)

AlexNet demonstrated breakthrough performance in image recognition.

The 21st century has witnessed a renaissance in AI, driven by the availability of big data and increased computing power. Geoffrey Hinton's work on deep learning in 2006 revolutionized neural network training. IBM's Watson showcased advanced natural language processing by winning Jeopardy! in 2011. AlexNet's success in the 2012 ImageNet competition demonstrated the power of deep learning for image recognition. DeepMind's AlphaGo marked a significant achievement in AI by defeating world Go champion Lee Sedol in 2014.

Harpreet Singh - Enterprise Security Architect



Key AI Approaches and Paradigms

Symbolic AI

Based on explicit rules and knowledge representation, strong in well-defined domains with clear rules.

Machine Learning

Algorithms improve through experience, dependent on patterns in data rather than explicit programming.

Deep Learning

Neural networks with multiple layers, effective in speech, vision, and language tasks.

AI encompasses various approaches, each with its strengths and limitations. Symbolic AI relies on explicit rules and knowledge representation, making it effective in well-defined domains. Machine learning algorithms improve through experience, leveraging patterns in data. Deep learning, using neural networks, excels in complex tasks such as speech, vision, and language processing. Hybrid approaches combine these paradigms to create more robust and interpretable systems.

Current State and Future Directions

AI continues to evolve rapidly, with key developments in foundation models, multimodal AI, and responsible AI. The field is moving towards systems that can process and generate multiple types of data, while addressing ethical concerns and ensuring alignment with human values. AI is increasingly integrated into everyday applications, promising a future where technology seamlessly enhances human capabilities. We must focus on responsible development and deployment to ensure these advancements benefit society as a whole.

Machine Learning: An Overview

Machine learning employs algorithms to discern patterns from data. Its goal is to enable systems to improve without explicit programming. The impact includes automation and data-driven decisions across various sectors. The market is projected to reach $211.71 billion by 2029.

Supervised Learning: Learning from Labeled Data

Definition

It trains a model using labeled input-output pairs. The model learns to map inputs to correct outputs.

- Spam detection
- Image classification
- Medical diagnosis

Algorithms

These are common supervised learning algorithms:

- Linear Regression
- Support Vector Machines (SVM)
- Decision Trees

Achieves 97% accuracy in image recognition with CNN models.

Supervised Learning: Classification

01
10

Binary Classification

Two outcomes (e.g., spam or not spam).

Multi-class Classification

More than two outcomes (e.g., classifying types of fruits).

Common metrics include precision, recall, F1-score, accuracy, and AUC-ROC. Use cases include diagnosing diseases and identifying objects in images.

Supervised Learning: Regression

1

Definition

Predicting continuous numerical values by modeling relationships.

2

Use Cases

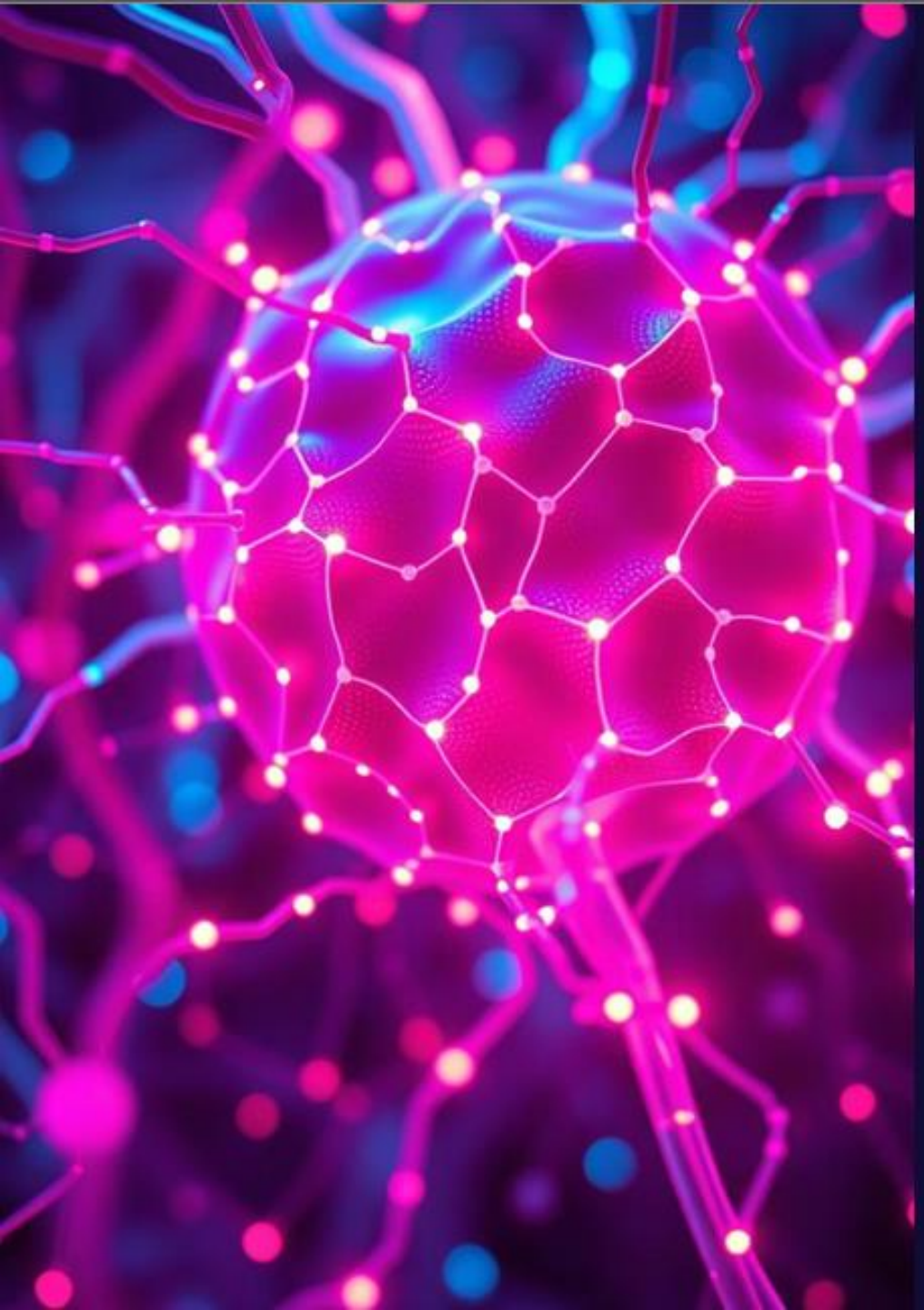
Sales forecasting, weather prediction, and stock market analysis.

3

Algorithms

Linear Regression, Polynomial Regression, and SVR.

Common metrics are Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). Predicting customer churn rate based on engagement metrics is an example.



Unsupervised Learning: Discovering Hidden Patterns

Definition

It trains a model using unlabeled data. It discovers inherent structure and relationships.

Algorithms

Clustering (K-Means, DBSCAN), Association Rule Learning, and Principal Component Analysis (PCA).

Use Cases

Customer segmentation, anomaly detection, and dimensionality reduction.

Recommending products based on purchase history is an example. Clustering techniques can improve customer retention by 30%.

Semi-Supervised Learning: Leveraging Labeled Data

1

Definition

It combines labeled and unlabeled data for training.

2

Rationale

It is cost-effective when labeling data is expensive.

3

Algorithms

Self-Training, Co-Training, and Label Propagation.

Use cases include speech, document, and image recognition. Provides 15-20% accuracy increase.

Reinforcement Learning: Learning Through Interaction

Definition

Trains an agent to make decisions in an environment to maximize a reward.

Process

Agent learns through trial and error, receiving rewards or penalties.

Algorithms

Q-Learning, Deep Q-Networks (DQN), and Policy Gradients.

Use cases include game playing, robotics, and resource management. AlphaGo is an RL system that beat the world champion in Go.

Deep Learning: Neural Networks with Many Layers

1

Definition

Uses neural networks with many layers to analyze data.

2

Key aspect

Automatically learns hierarchical representations of data.

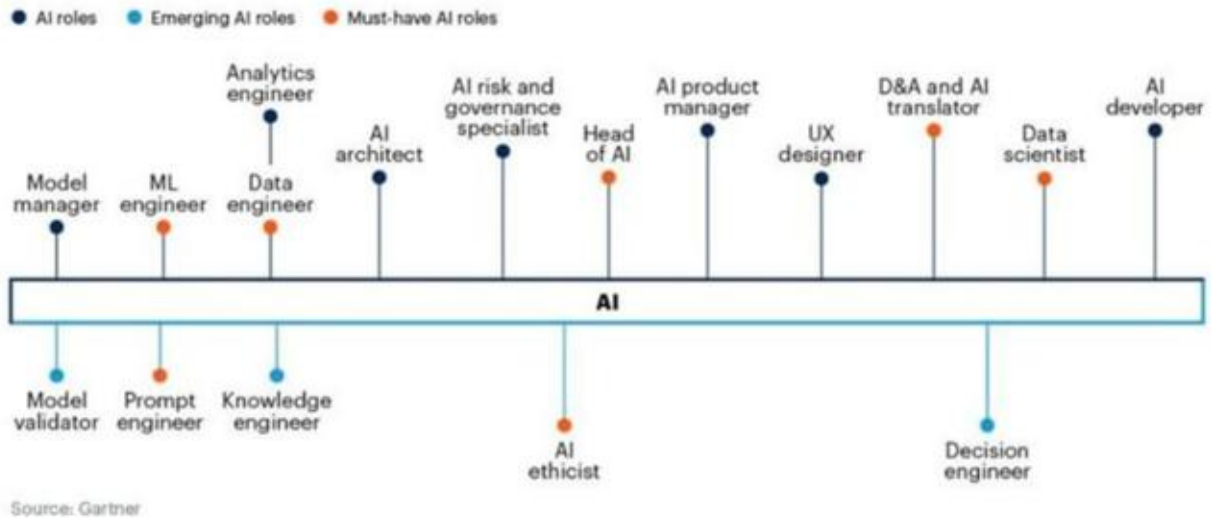
3

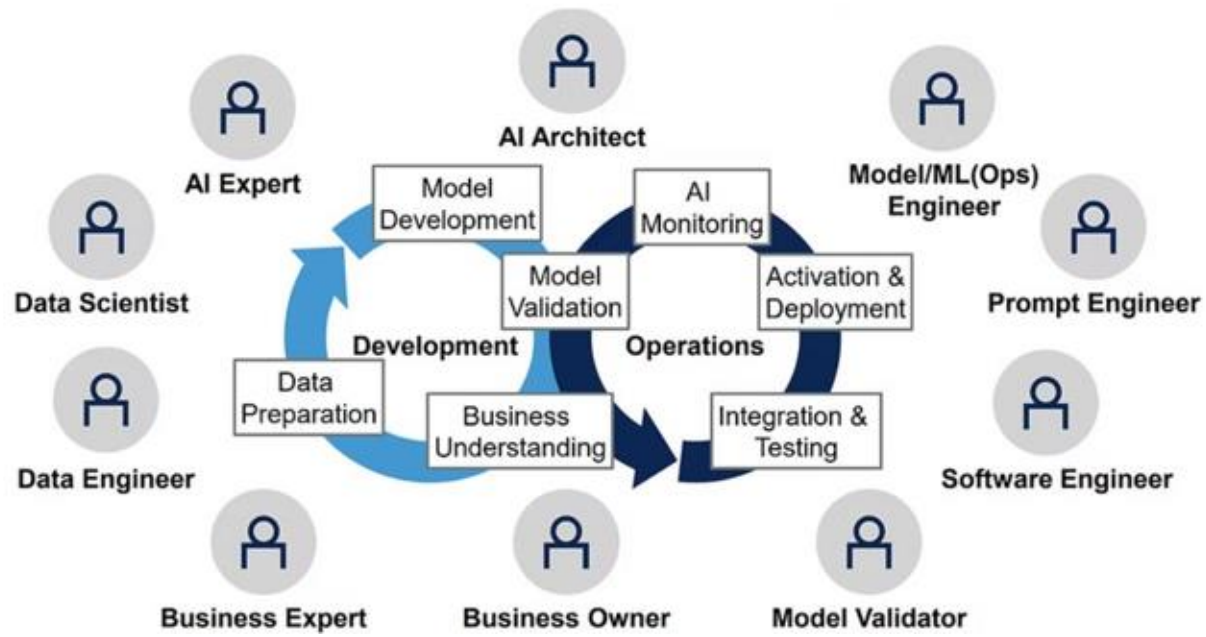
Architectures

CNNs, RNNs, and Transformers.

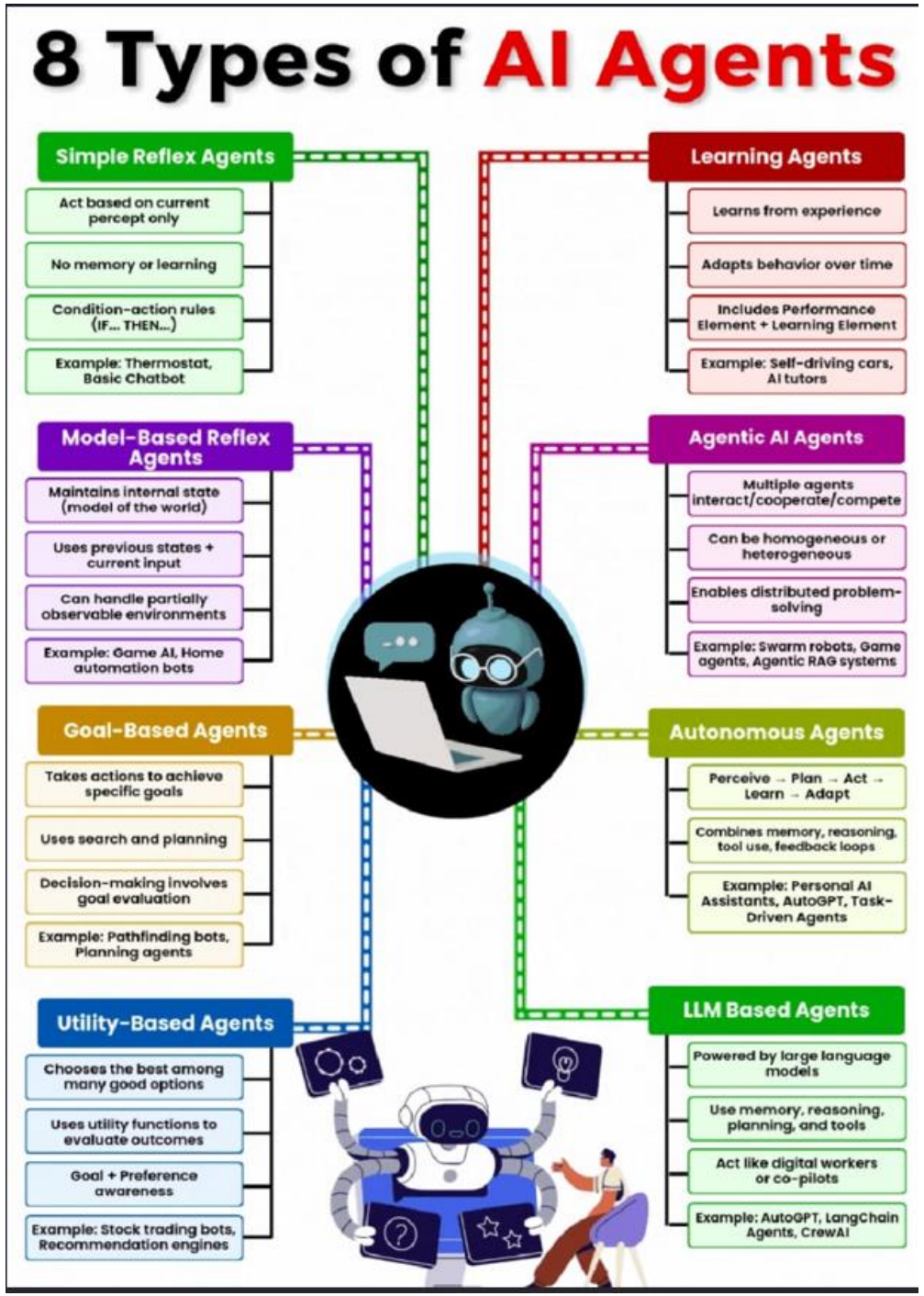
It has revolutionized computer vision and NLP. It has achieved human-level performance on tasks like image classification.

New AI Roles





Source - Gartner



Harpreet Singh - Enterprise Security Architect

Securing AI Systems

What is Artificial Intelligence

Artificial Intelligence (AI) is a broad field within computer science focused on creating machines and computer programs that can perform tasks that typically require human intelligence. These tasks include:

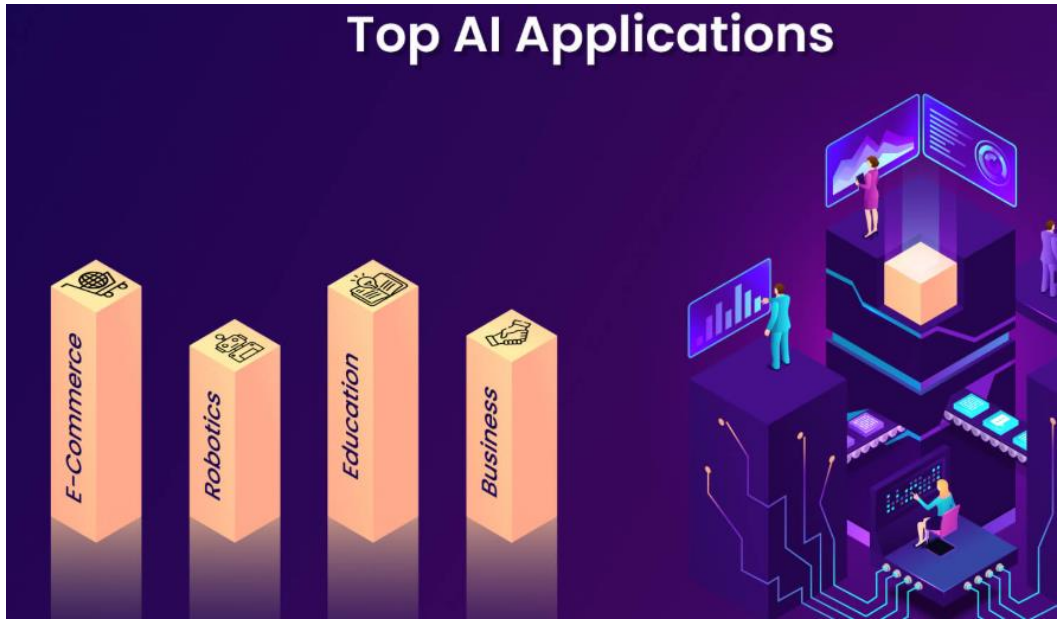
- **Learning:** Acquiring information and rules for using the information.
- **Reasoning:** Using rules to reach conclusions.
- **Problem-solving:** Finding solutions to issues.
- **Perception:** Understanding sensory input (like vision, sound, and language).
- **Language understanding:** Processing and comprehending human language.
- **Decision-making:** Choosing the best course of action.
- **Creativity:** Generating novel and valuable ideas or artifacts.

In simpler terms, AI aims to make computers "think" and act intelligently, similar to humans

Potential Application of AI

AI is rapidly transforming various industries and aspects of our lives, including:

- **Healthcare:** Disease diagnosis, drug discovery, personalized medicine.
- **Finance:** Fraud detection, algorithmic trading, risk assessment.
- **Transportation:** Self-driving vehicles, traffic management.
- **Education:** Personalized learning, intelligent tutoring systems.
- **Retail:** Personalized recommendations, chatbots for customer service.
- **Manufacturing:** Automation, quality control, predictive maintenance.
- **Entertainment:** Content recommendation, game AI.
- **Security:** Threat detection, facial recognition.
- **Everyday Life:** Virtual assistants, smart home devices, spam filtering.



The field of AI is constantly evolving, with new breakthroughs and applications emerging regularly. It holds immense potential to solve complex problems and improve our lives, but it also raises important ethical and societal considerations that need careful attention.

Artificial Intelligence (AI):

- AI is a broad field of computer science focused on creating machines that can perform tasks that typically require human intelligence.
- This includes abilities like:
    - Learning
    - Problem-solving
    - Understanding language
    - Recognizing patterns

Large Language Models (LLMs):

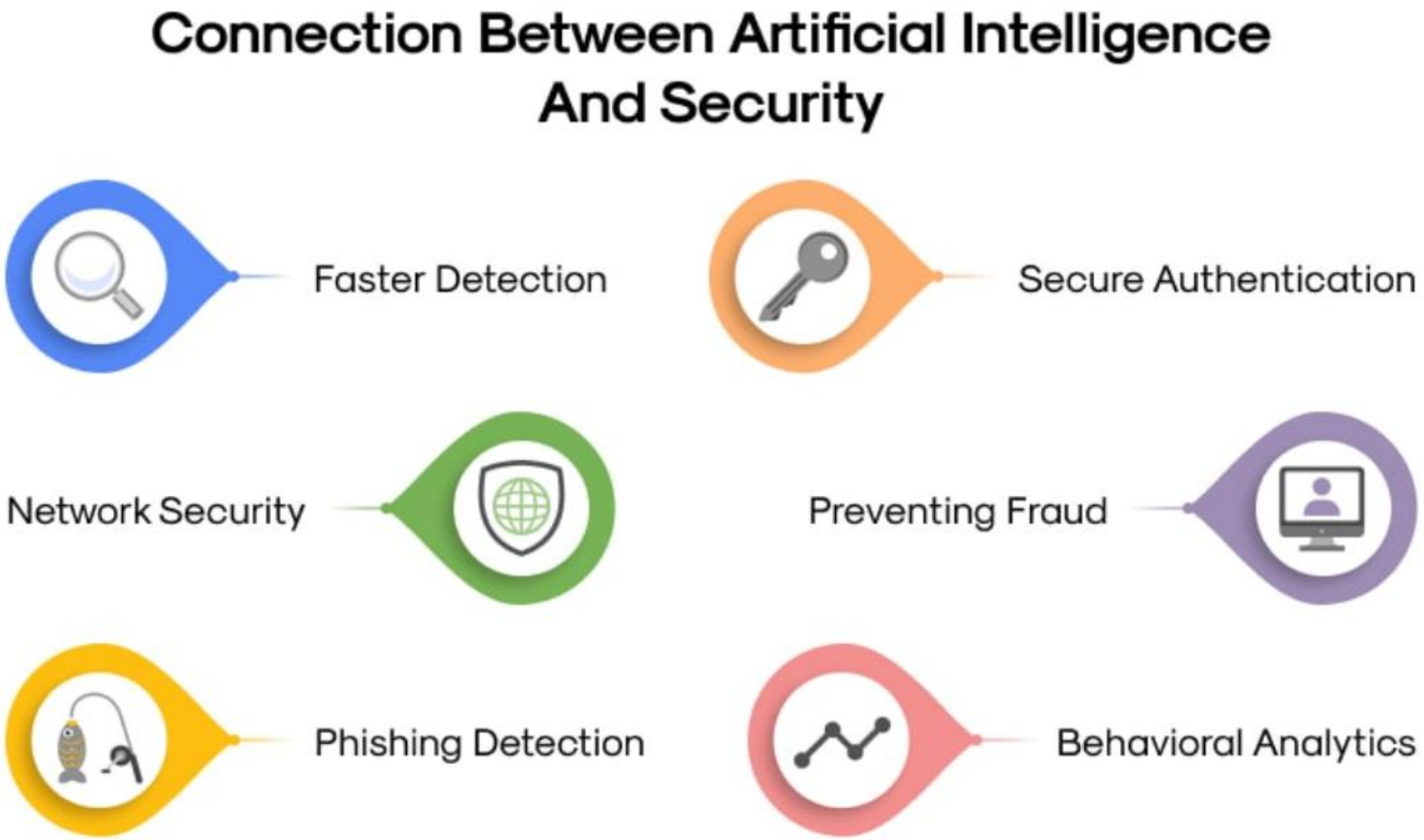
- LLMs are a specific type of AI model.
- They use deep learning techniques to understand and generate human-like text.
- They are "large" because they are trained on massive datasets of text and code.
- LLMs are a subset of generative AI, which is a portion of the broader AI field.

How LLMs Relate to AI:

- LLMs are a significant advancement within the field of AI, particularly in the area of natural language processing (NLP).
- They demonstrate AI's ability to:
    - Comprehend complex language structures.
    - Generate coherent and contextually relevant text.
    - Perform tasks like translation, summarization, and question answering.
- Essentially, LLMs are a tool that brings AI closer to effectively interacting with humans through natural language.
- LLMs are a component of generative AI, which is a branch of AI that focuses on creating new content.

How does AI relate to LLM ?

Harpreet Singh - Enterprise Security Architect



Why is AI Security required at all ?

Harpreet Singh - Enterprise Security Architect

Protecting AI Powered Security System

- Ensure the AI Powered Systems are not compromised otherwise the ability to Identify, Protect, Detect the attacks can be eroded leading to false positives, undetected attacks, compromised security posture
- Adversarial inputs that can trick AI models into misclassifying malicious activity as benign would be used for malicious intents like poisoning, spoofing of AI
- Ensure that the data used by AI-driven security tools remains untampered and reliable hence any AI based Data should not be prone to Data Poisoning causing AI Models to be biased

Securing AI System – Creating AI Defense in Depth

- **Preventing Unauthorized Access and Data Breaches:** AI models and the data they are trained on can be highly valuable assets, containing sensitive information or proprietary algorithms. AI security measures are crucial to protect these assets from unauthorized access, theft, and data breaches.

- Mitigating Adversarial Attacks: AI systems are vulnerable to adversarial attacks, where carefully crafted malicious inputs can cause them to malfunction or produce incorrect outputs. This can have severe consequences, especially in critical applications like autonomous vehicles or medical diagnosis.

- Protecting Against Model Theft and Reverse Engineering: Proprietary AI models represent significant investments in research and development. AI security aims to prevent attackers from stealing or reverse-engineering these models, which could lead to intellectual property loss and competitive disadvantages

Harpreet Singh - Enterprise Security Architect

Ensuring the Integrity of AI Pipelines: The lifecycle of an AI model involves various stages, from data collection and training to deployment and monitoring. Security is needed at each stage to prevent malicious interference that could compromise the model's integrity and performance.

Addressing New Threats Introduced by AI: AI technologies, particularly generative AI, can be misused to create sophisticated phishing attacks, deepfakes, and automated malware. AI security is required to develop defenses against these novel AI-powered threats.

Maintaining User Trust and Confidence: As AI becomes more integrated into everyday services, ensuring its security is vital for maintaining user trust and confidence in these technologies. Security breaches or unreliable AI outputs can erode this trust.

Complying with Regulations: With increasing awareness of the potential risks associated with AI, governments and regulatory bodies are beginning to implement data governance and AI-specific regulations. AI security measures will be essential for organizations to comply with these evolving legal requirements

- **Artificial Intelligence (AI):**
  - This is the broadest category. AI refers to the simulation of human intelligence in machines. It's the concept of creating machines that can perform tasks that typically require human intelligence.
  - AI encompasses a wide range of techniques, including rule-based systems, expert systems, and machine learning.
- **Machine Learning (ML):**
  - ML is a subset of AI. It focuses on developing algorithms that allow computers to learn from data without explicit programming.
  - Instead of being explicitly told how to perform a task, ML algorithms learn patterns from data and use those patterns to make predictions or decisions.
- **Natural Language Processing (NLP):**
  - NLP is a subset of AI, and often, a subset of ML. It's the field that deals with enabling computers to understand, interpret, and generate human language.
  - NLP techniques are used in various applications, such as chatbots, language translation, and sentiment analysis.
- **Large Language Models (LLMs):**
  - LLMs are a subset of ML, and a specialized application of NLP. They are AI models that have been trained on massive amounts of text data, enabling them to understand and generate human-like text.
  - LLMs are a powerful tool within NLP, significantly advancing the field's capabilities.

The Greatest Nexus – AI, ML, LLM, NLP

Harpreet Singh - Enterprise Security Architect

Here's a simplified way to visualize their relationship:

- AI: The overarching goal of creating intelligent machines.
- ML: A technique for achieving AI through data-driven learning.
- NLP: A field within AI focused on processing human language.
- LLMs: A type of ML model that excels at NLP tasks

AI	Artificial Intelligence	The big umbrella – any machine intelligence
NLP	Natural Language Processing	A subfield of AI focused on understanding and using human language
LLM	Large Language Model	A type of AI model built using NLP methods to generate and understand language

The Greatest Nexus – AI, ML, LLM, NLP

Harpreet Singh - Enterprise Security Architect

- **Artificial intelligence** (AI) is a broad term that encompasses all fields of computer science that enable machines to accomplish tasks that would normally require human intelligence. Machine learning and generative AI are two subcategories of AI.

- **Machine learning** (ML) is a subset of AI that focuses on creating algorithms that can learn from data. Machine learning algorithms are trained on a set of data, and then they can use that data to make predictions or decisions about new data.

- **Generative AI** is a type of machine learning that focuses on creating new data. Often, GenAI relies on the use of large language models to perform the tasks needed to create the new data.

- A **large language model (LLM)** is a type of AI model that processes and generates human-like text. In the context of artificial intelligence a "model" refers to a system that is trained to make predictions based on input data. LLMs are specifically trained on large data sets of natural language and the name large language models.

The Greatest Nexus – AI, ML, GEN AI, LLM

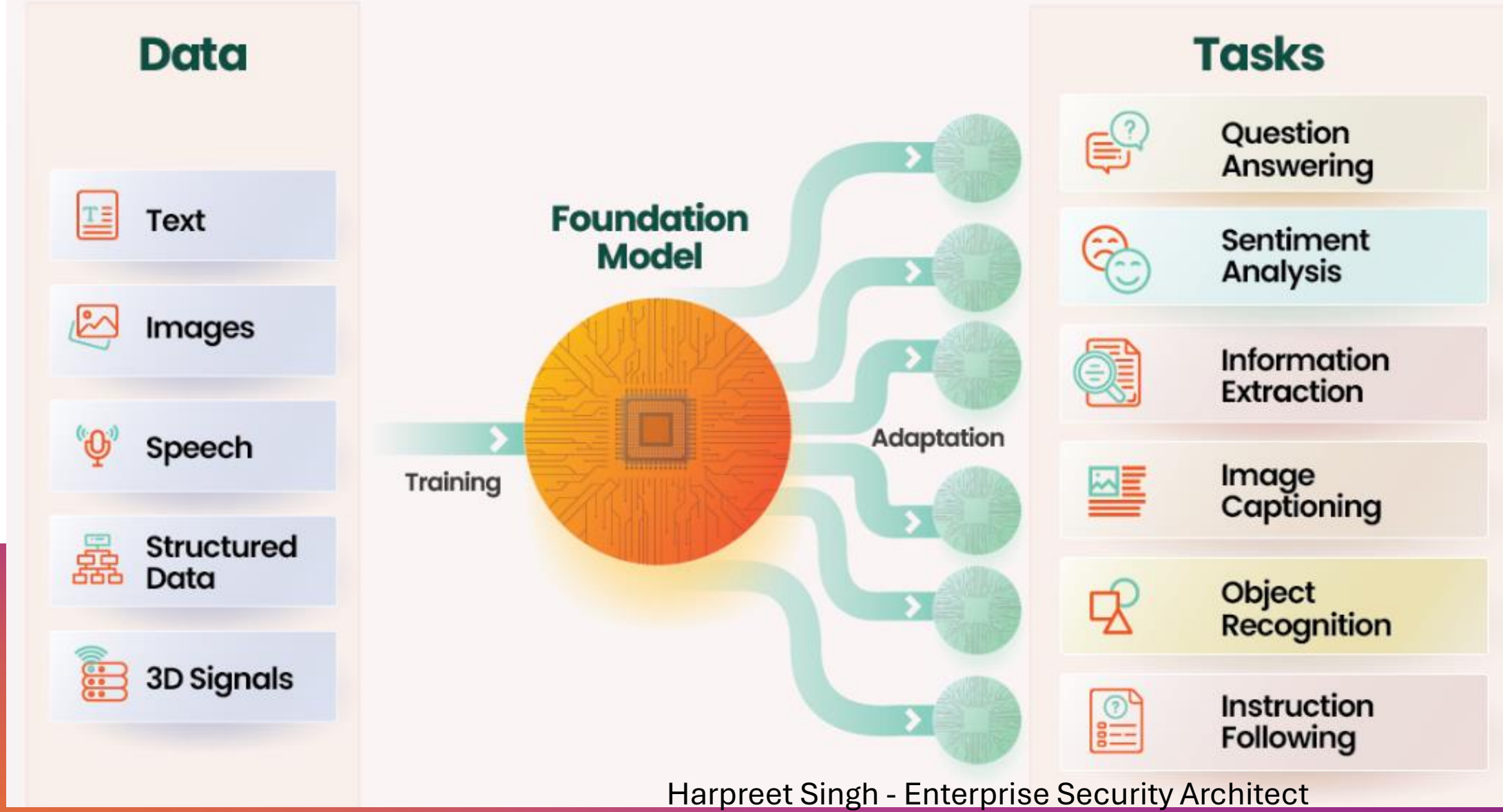
Harpreet Singh - Enterprise Security Architect

Gen AI – What's the buzz about

Generative AI, often shortened to Gen AI, is a branch of artificial intelligence focused on creating new content. Unlike traditional AI that primarily analyses or categorizes existing data, generative AI models can produce original outputs. These outputs can take various forms, including:

- **Text:**
  - Writing articles, poems, code, and even scripts.
- **Images:**
  - Generating realistic or artistic images from text descriptions.
- **Audio:**
  - Creating music, speech, and sound effects.
- **Video:**
  - Producing animated or realistic video content.
- **Code:**
  - Creating computer code in various programming languages.
- **And more:**
  - Including 3D models, and even chemical formulas.

How does Generative AI work?



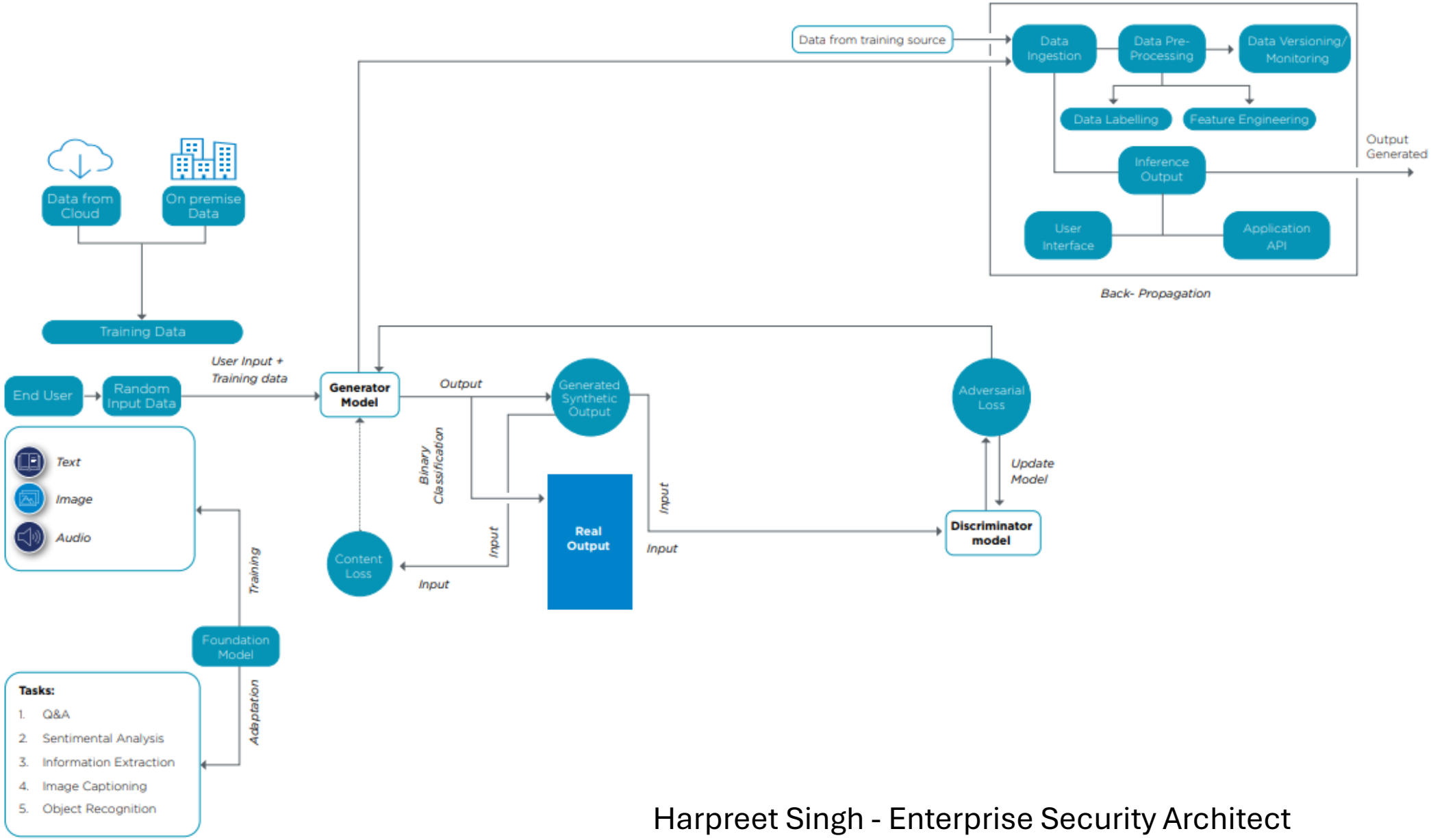
Gen AI – How does it work

How Generative AI Works

Generative AI works through a process of pattern recognition, learning from vast datasets, and then generating new content based on what it's learned. Here's a simplified explanation:

1. **Training on Large Datasets**: Models like me (Claude) are trained on enormous amounts of text from the internet, books, articles, and other sources.
2. **Neural Network Architecture**: We use complex neural networks—specifically transformer architectures—that learn to understand language patterns.
3. **Pattern Recognition**: During training, the model learns to recognize patterns in language, how words relate to each other, factual information, and various writing styles.
4. **Probability-Based Prediction**: When generating text, the model predicts what words or tokens are most likely to come next in a sequence, based on what it learned during training.
5. **Parameter Tuning**: Models have billions of parameters (weights in the neural network) that are adjusted during training to improve performance.
6. **Fine-Tuning and Alignment**: After initial training, models undergo additional training to better align with human values, follow instructions, and avoid generating harmful content.

The Architecture of Generative AI Technology



Harpreet Singh - Enterprise Security Architect

The architecture can be broadly divided into three main sections:

1. **Data Acquisition and Model Training (Left and Bottom Sections)**
2. **Generative Adversarial Network (GAN) Core (Middle Section)**
3. **Deployment and Operationalization (Top Right Section)**

---

1. Data Acquisition and Model Training

At the foundation of this architecture is the data used for training and the underlying models.

- **Data Sources:**
  - **Data from Cloud:** Represents training data residing in cloud storage solutions.
  - **On-premise Data:** Represents training data stored locally within an organization's infrastructure.
  - These two sources feed into a central **Training Data** component.
- **Foundation Model:**
  - The **Training Data** is used to train a **Foundation Model**. This suggests a large pre-trained model capable of understanding and generating various types of data.

- The diagram shows "Training" from the Training Data to the Foundation Model.
- "Adaptation" from the Foundation Model points to the Text, Image, and Audio inputs, indicating that the Foundation Model can adapt to or process different modalities of data.
- **Tasks:** The Foundation Model is equipped to handle a range of Generative AI tasks, including:
    1. Q&A (Question & Answer)
    2. Sentimental Analysis
    3. Information Extraction
    4. Image Captioning
    5. Object Recognition

2. Generative Adversarial Network (GAN) Core

This section illustrates the core components of a GAN, which involves two neural networks, a Generator and a Discriminator, competing against each other.

- **End User Interaction & Input:**
  - An **End User** provides **Random Input Data**, which can be Text, Image, or Audio. This "Random Input Data" combined with "Training data" forms the "User Input + Training data" that feeds into the Generator Model. This suggests a conditional GAN, where the generator's output is conditioned on user input.
- **Generator Model:**
  - The **Generator Model** takes "User Input + Training data" as its primary input.
  - Its function is to learn the distribution of the real data and generate new, synthetic data that is indistinguishable from real data.
  - It produces an "Output" which leads to **Generated Synthetic Output**.

- **Discriminator Model:**
  - The **Discriminator Model** acts as a binary classifier. Its job is to distinguish between "Real Output" (actual data from the training set) and "Generated Synthetic Output" (data produced by the Generator).
  - It takes "Input" from both "Generated Synthetic Output" and "Real Output" (represented by the blue box).
  - The Discriminator performs "Binary Classification" on its inputs, attempting to identify if the data is real or fake.
  - **Adversarial Loss:** The Discriminator's performance feeds back as "Adversarial Loss" to the Generator. This loss guides the Generator to produce more realistic outputs that can fool the Discriminator. An arrow from "Adversarial Loss" points to "Update Model" for the Discriminator, indicating that the Discriminator's parameters are updated based on its classification accuracy.

- **Feedback Loops for Training:**
  - **Content Loss:** An additional "Content Loss" is depicted, which takes "Input" from the Generator Model's output. This suggests that beyond the adversarial training, there might be a perceptual or content-based loss function applied to ensure the generated output is semantically meaningful or aligns with the input in specific ways. This "Content Loss" feeds back directly to the "Generator Model", helping it refine the quality of its generated content.
  - The interaction between the Generator and Discriminator is iterative: The Generator tries to create better fakes, and the Discriminator tries to become better at spotting them. This continuous training refines both models.

3. Deployment and Operationalization

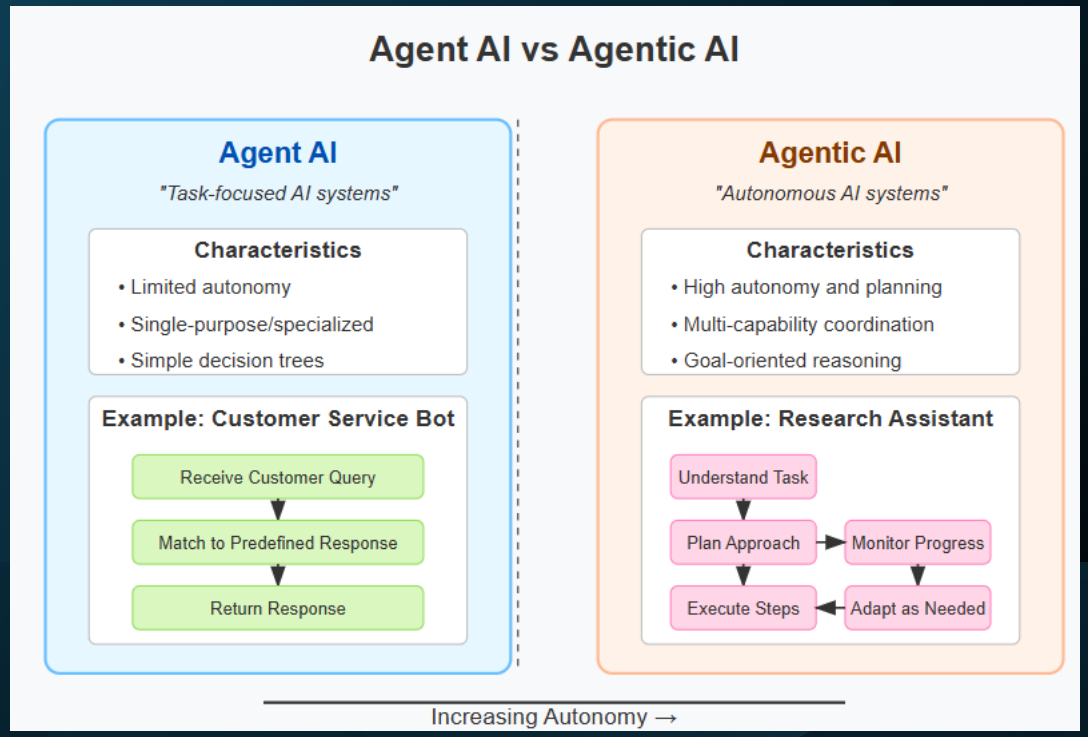
This section outlines the pipeline for taking generated output and making it available for applications and users. This typically represents the inference or production environment for the Generative AI model.

- **Data from Training Source:** This arrow indicates that data from the original training source is also leveraged in the production pipeline, likely for processes like monitoring and quality control.

- **Data Processing Pipeline (Output Generated):**
  - **Data Ingestion:** The initial step, where data relevant to the inference process is collected.
  - **Data Pre-processing:** Preparing the ingested data for the model, which might include normalization, tokenization, or resizing.
  - **Data Versioning/Monitoring:** Ensuring data consistency and tracking data quality over time. This is crucial for maintaining model performance and detecting drift.
  - **Data Labelling & Feature Engineering:** These steps, though typically part of initial model development, are shown here in the deployment pipeline, suggesting ongoing processes for refining or re-training, or for monitoring data characteristics of the generated output.
  - **Inference Output:** This is the core step where the Generative AI model produces its final outputs based on inputs.

- **Access and Integration:**
    - **User Interface:** A graphical interface allowing end-users to interact with the Generative AI system and receive its "Output Generated".
    - **Application API:** An Application Programming Interface, allowing other applications or services to programmatically interact with the Generative AI model and retrieve its "Output Generated".
- **Back-Propagation:** The label "Back-Propagation" underneath the deployment pipeline box implies that feedback from the generated output (e.g., user ratings, performance metrics, or identified issues) can be used to improve the upstream models (Generator, Foundation Model). This signifies a continuous improvement loop, not just during initial training but also in production.

Agent AI vs Agentic AI – How does it work



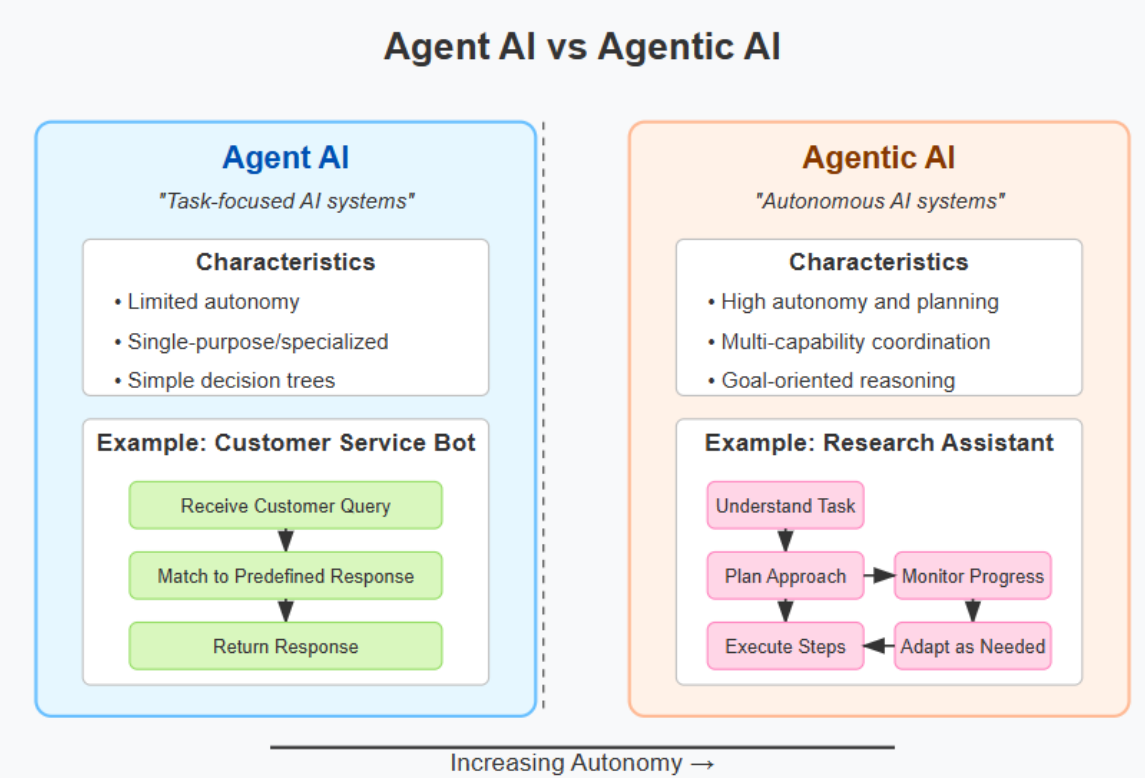
- **Agent AI vs Agentic AI**
- While the terms "Agent AI" and "Agentic AI" are often used interchangeably, there are subtle distinctions in how they're typically understood in the AI community.

- **Core Differences**
- **Agent AI:**
  - Typically refers to any AI system that can act in an environment
  - May follow simple rules or scripts
  - Often involves a single, specialized function
  - Usually operates with limited autonomy

- **Agentic AI:**
  - Refers to a more advanced form of AI with greater autonomy
  - Can plan, reason, and execute complex multi-step tasks
  - Often involves a system that can chain together multiple capabilities
  - Makes decisions based on goals and can adapt to changing circumstances

Harpreet Singh - Enterprise Security Architect

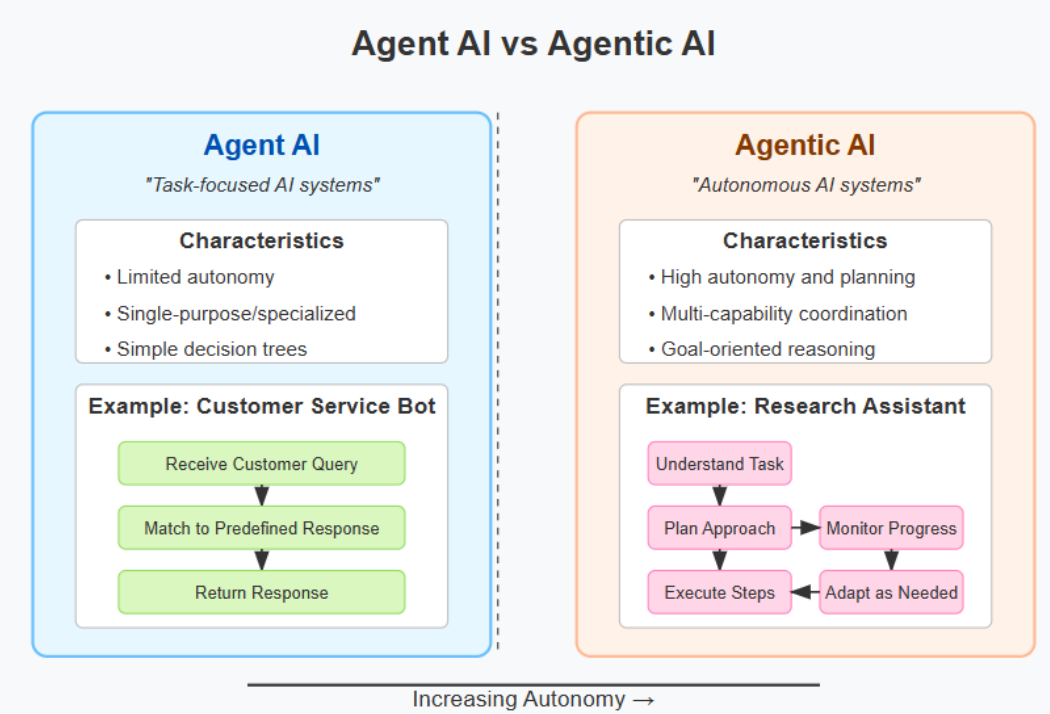
Agent AI vs Agentic AI – How does it work



- **Key Aspects of Each Type**
- **Agent AI**
  - **Structure**: Typically follows a predefined, linear workflow
  - **Capabilities**: Usually has specialized knowledge or skills in one domain
  - **Decision-making**: Often based on rule-based systems or simple condition checks
  - **Examples**: Simple chatbots, automated customer service systems, specific task automations
- **Agentic AI**
  - **Structure**: Features complex, interconnected systems that work together
  - **Capabilities**: Integrates multiple AI skills (reasoning, planning, tool use)
  - **Decision-making**: Can formulate plans, evaluate approaches, and adjust strategies
  - **Examples**: AI research assistants, autonomous workflow systems, "AI agents" like Claude Code

Harpreet Singh - Enterprise Security Architect

Agent AI vs Agentic AI – How does it work



Evolution of AI Agents

The distinction represents an evolution in AI capabilities:

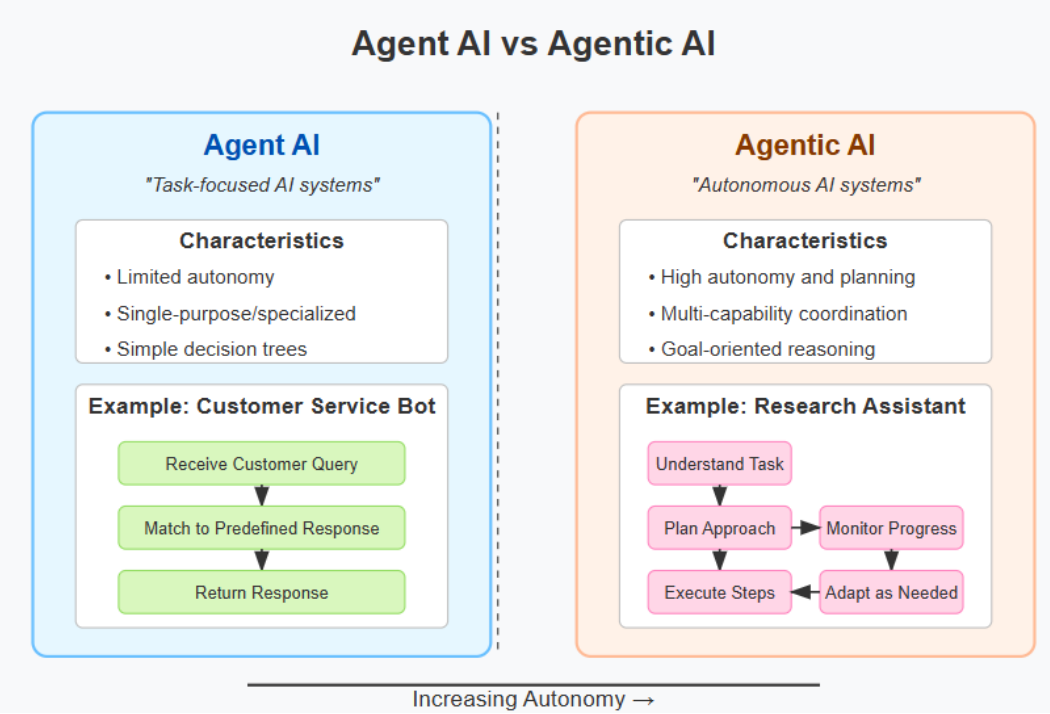
1. **Traditional Agent AI**: Follows scripts and rules with minimal adaptation
2. **Early LLM Agents**: Can handle natural language but with limited planning
3. **Modern Agentic AI**: Combines LLM reasoning with planning, tool use, and feedback loops
4. **Future Agentic Systems**: Will likely feature even greater autonomy and self-improvement

Common Agentic AI Components

1. **Planning**: Creating step-by-step approaches to solve complex problems
2. **Memory**: Retaining and utilizing context from previous interactions

Harpreet Singh - Enterprise Security Architect

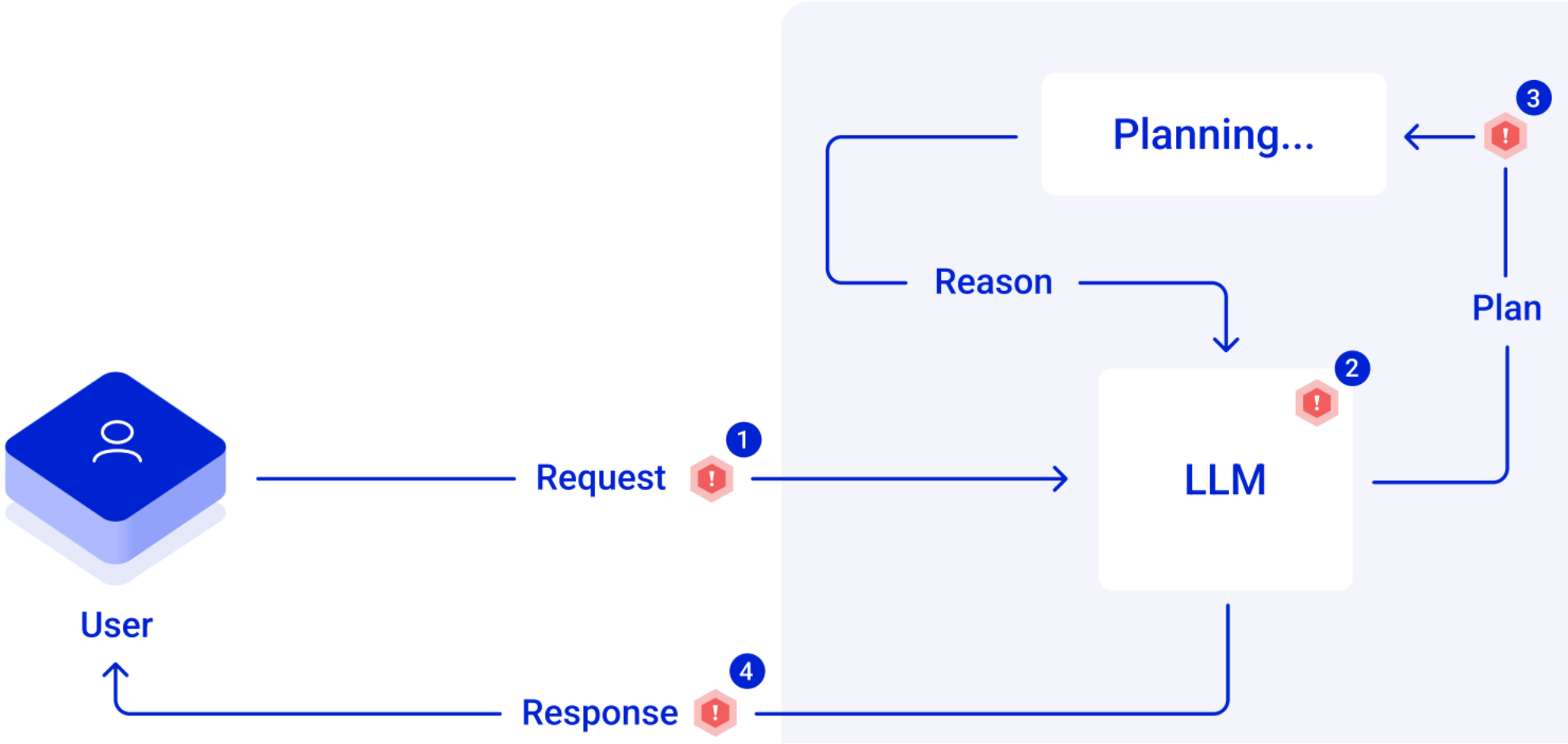
Agent AI vs Agentic AI – How does it work



3. Tool Use: Leveraging external tools like code execution, database access, etc.

4. Self-reflection: Evaluating performance and adjusting strategies accordingly

5. Multi-agent collaboration: Coordinating between specialized sub-agents



LLM Chatbot Reference Architecture with Threats

Harpreet Singh - Enterprise Security Architect

Common threats to chatbots arise when there is

- (1) **untrusted input** – input is poisoned, malicious or not from a trusted source
- (2) **a misaligned model** (for example, through fine-tuning) – when alignment for intended outcome is not as per reference architecture
- (3) **prompt injection to override or extract the system prompt** – inserting invalid statement to override or extract the proper prompt that accounts for an output
- (4) **unvalidated output** – Expected output as a result does not get validated per modelled workflow

LLM Chatbot Reference Architecture

Harpreet Singh - Enterprise Security Architect

Types of Chatbot

Single-purpose chatbots

Single-purpose chatbots are designed to excel in specific domains or tasks, such as customer support, booking systems, or FAQ automation. Examples include:

- Customer support chatbots for e-commerce
- Booking assistants for hotels and flights
- Educational tutors for specific subjects

Hybrid chatbots

Hybrid chatbots combine rule-based and AI-driven approaches to handle both predictable and complex interactions effectively. Examples include:

- Retail chatbots that offer standard shopping assistance while handling complex customer queries
- Health advisory chatbots that provide generic information and tailored medical consultations

Context-aware chatbots

Context-aware chatbots use memory capabilities to remember past interactions, thereby providing more personalized and coherent responses. Examples include:

- Personal assistant bots that manage schedules and preferences
- Finance advisory bots that track user transactions and provide tailored advice

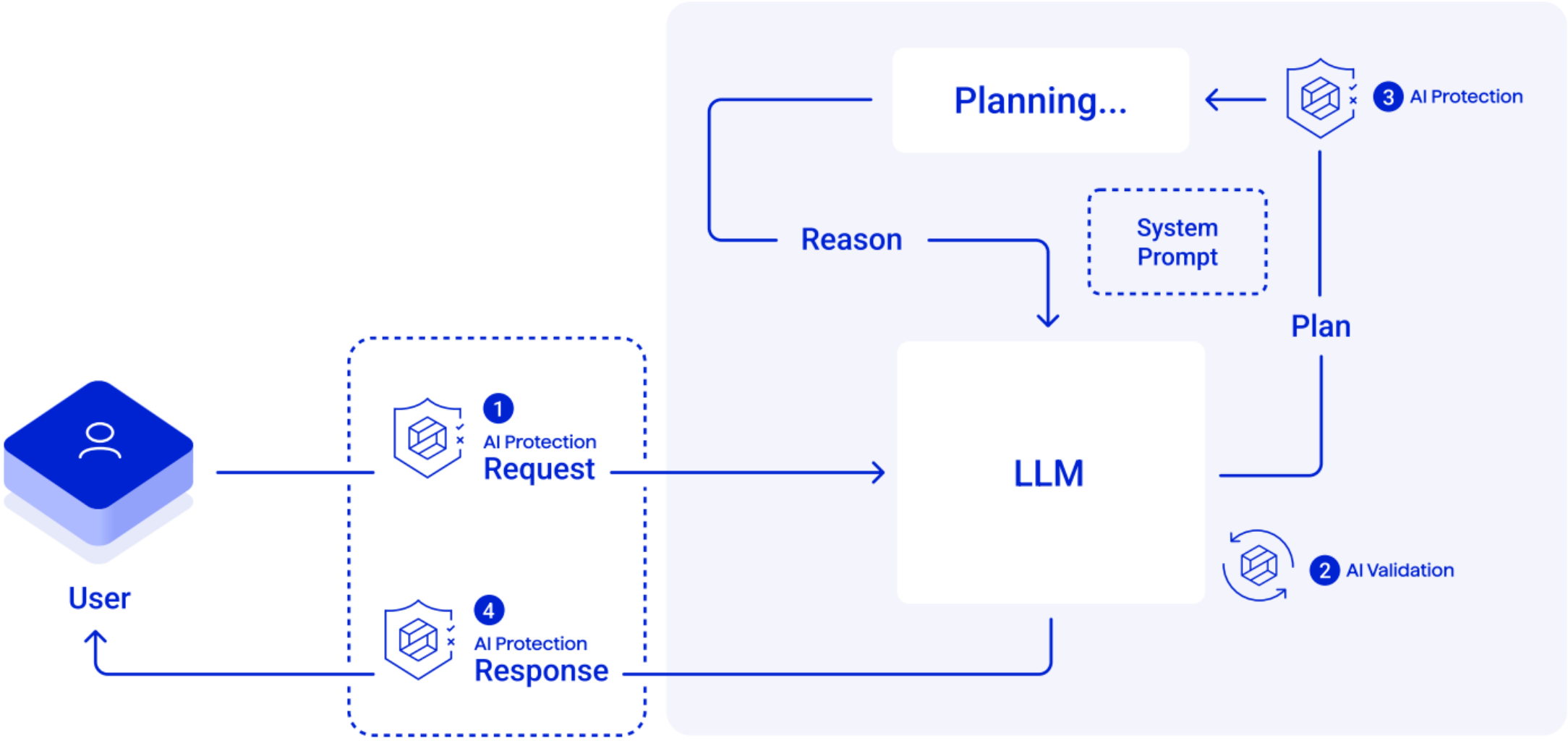
Key Technologies in Chatbot Design - What does it include ?

- LLM models
- Foundational development frameworks.

For example:
https://github.com/run-llama/llama_index
https://github.com/cpacker/MemGPT
https://github.com/ollama/ollama
https://github.com/neuml/txtai

- Vector databases, including: - FAISS, HNSW, ChromaDB, Pinecone, LanceDB, Qdrant, and Weaviate
- Prompt engineering
- Model fine-tuning, which can be done with tools such as:
- OpenAI fine-tuning service
- Azure OpenAI
- Together.ai

Non-Threats based Architecture for Chatbot

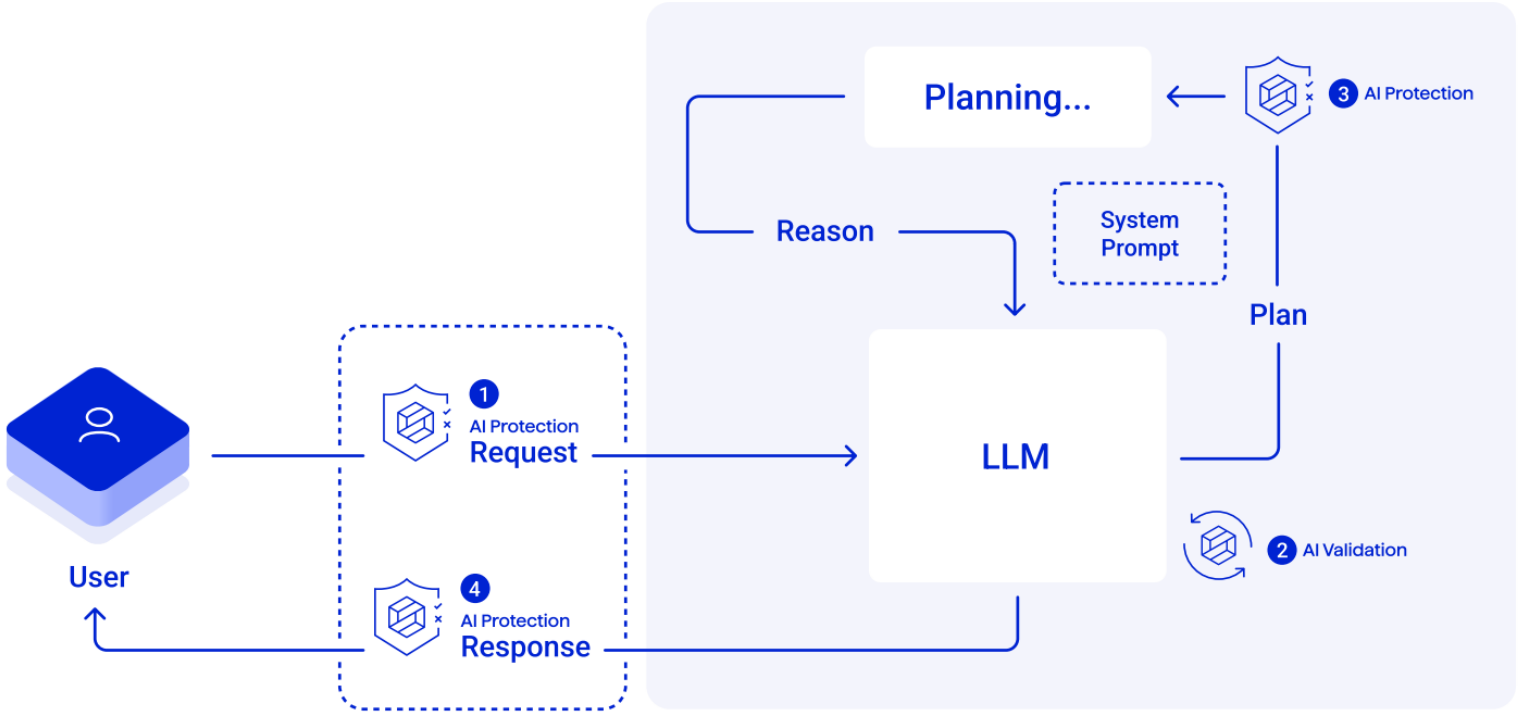


Harpreet Singh - Enterprise Security Architect

Non-Threat based Architecture for Chatbot

Summary on the points

- The LLM (2) should be validated for alignment, safety, and security before selection and after fine-tuning.
- At runtime, requests (1) and responses (4) should include real-time protection to prevent attempts to misuse/abuse the application, as well as any unsafe output.
- Proper request filtering can (3) neutralize attempts at prompt injection and prompt extraction



Harpreet Singh - Enterprise Security Architect

Mitigation measure that be proactively looked into while designing & implementing

User interaction threats

Chatbots are typically exposed to a large user population and must be protected from malicious user actions.

- **Injection and Jailbreak Attacks:** Malicious users could input specially crafted messages attempting to manipulate the chatbot's responses or extract sensitive data.
  - **Mitigations:**
    - Implement input validation and sanitization to detect and block harmful inputs before they can affect the chatbot's operation.
    - Implement output filtering to prevent harmful or potentially malicious LLM responses from being returned to the user
    - Implement strong system prompt practices to strictly scope the application to its intended use case.
    - Implement additional off-topic enforcement to ensure the application stays within scope.
    - Limit the attack surface that can accept external prompt sources

Mitigation measure that be proactively looked into while designing & implementing

- **Denial of Service (DoS):** A user or bot could overwhelm the chatbot with a flood of messages, causing it to become unresponsive.
  - **Mitigations:**
    - Employ rate limiting on incoming messages to manage and mitigate excessive traffic
    - Implement auto-scaling and resource allocation strategies to ensure the system can handle spikes in demand without degradation of service.
    - Implement input validation to detect and block adversarial attacks that lead to denial of service.

Mitigation measure that be proactively looked into while designing & implementing

User interaction threats

LLMs are trained on a large amount of data. This data must be kept in alignment with the organization's ethics, and the LLM must not divulge data that's inappropriate for a user.

- **Model Poisoning:** The chatbot could be trained on malicious input over time, leading it to make incorrect or offensive statements.
  - **Mitigations:**
    - Implement mechanisms to identify prompts containing harmful, toxic, or malicious content prior to entering the training pipeline.
    - Regularly monitor and audit the training data for harmful, toxic, or adversarial inputs that may cause drift or introduce vulnerabilities.
    - Use robust fine-tuning practices to maintain the integrity of the model's responses.

Mitigation measure that be proactively looked into while designing & implementing

- **Exfiltration from ML Application:** If the LLM has been trained on sensitive data, users might manipulate the LLM to reveal that data in its responses.
  - **Mitigations:**
    - Use output filtering and moderation to prevent sensitive data from being included in the chatbot's responses.
    - Ensure strict access controls and authentication mechanisms are in place to safeguard sensitive information.
    - Implement the principle of least privilege to limit the availability of sensitive data

Mitigation measure that be proactively looked into while designing & implementing

User interaction threats

If an LLM has the ability to store information from its conversations, this information must be protected.

- **Unauthorized Access:** Attackers could gain access to long-term memory stores, compromising user privacy and data security.
  - **Mitigations:**
    - Implement comprehensive access controls and encryption to protect memory content.
    - Regularly update and patch storage systems to mitigate vulnerabilities that could be exploited for unauthorized access.

Mitigation measure that be proactively looked into while designing & implementing

- **Data Corruption:** Memory content could be altered or corrupted, leading to incorrect responses by the chatbot.
  - **Mitigations:**
    - Use redundancy and backup strategies to maintain the integrity and availability of data.
    - Implement data validation mechanisms to detect and correct any corruption
    - or unauthorized modifications.

What is RAG - How does it work

Retrieval Augmented Generation (RAG)

Retrieval Augmented Generation (RAG) is an AI technique that enhances language model outputs by combining the generative capabilities of large language models (LLMs) with information retrieved from external knowledge sources. This approach helps overcome knowledge limitations and improves factual accuracy.

How RAG Works

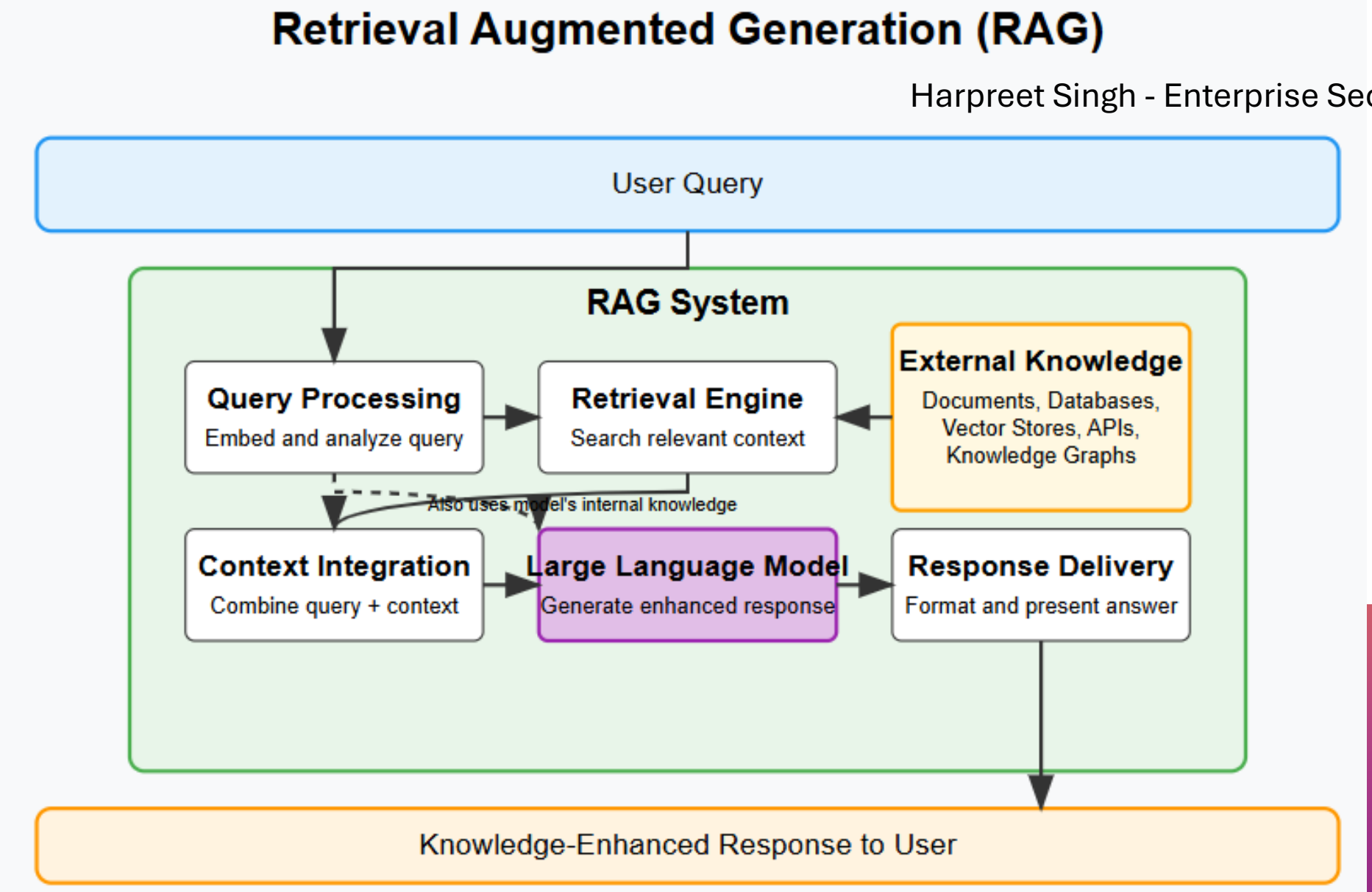
1. **Query Processing**: The user's query is processed and understood
2. **Information Retrieval**: Relevant information is retrieved from external knowledge sources (documents, databases, APIs)
3. **Context Integration**: The retrieved information is combined with the original query
4. **Enhanced Generation**: The LLM generates a response using both its internal knowledge and the retrieved context
5. **Response Delivery**: The final answer is provided to the user

Harpreet Singh - Enterprise Security Architect

Key Benefits of RAG

1. **Factual Accuracy**: Provides up-to-date, verifiable information from trusted sources
2. **Knowledge Expansion**: Extends beyond the LLM's training data cutoff
3. **Reduced Hallucinations**: Grounds responses in retrieved facts rather than generating information from memory
4. **Transparency**: Can cite sources for information retrieved
5. **Customization**: Can be tailored to specific knowledge domains

What is RAG - How does it work



What is RAG - How does it work

Key Benefits of RAG

1. **Factual Accuracy**: Provides up-to-date, verifiable information from trusted sources
2. **Knowledge Expansion**: Extends beyond the LLM's training data cutoff
3. **Reduced Hallucinations**: Grounds responses in retrieved facts rather than generating information from memory
4. **Transparency**: Can cite sources for information retrieved
5. **Customization**: Can be tailored to specific knowledge domains

What is RAG – how does it help LLM

Retrieval-Augmented Generation (RAG) is a technique that enhances the capabilities of Large Language Models (LLMs) by allowing them to access and incorporate information from external knowledge sources. Here's a breakdown:

The Problem with LLMs:

- **Limited Knowledge:** LLMs are trained on vast datasets, but their knowledge is limited to the data they were trained on. This means they can't provide accurate information on recent events or domain-specific knowledge they haven't encountered.
- **"Hallucinations":** LLMs can sometimes generate inaccurate or fabricated information, known as "hallucinations," because they're designed to produce text that sounds plausible, even if it's not factual.

Retrieval-Augmented Generation (RAG) with Language Models (LLMs) represents an advanced approach in conversational AI. It enhances the LLM's capacity by integrating external data retrieval into the response generation process

Harpreet Singh - Enterprise Security Architect

What is RAG – how does it help LLM

How RAG Solves These Problems:

RAG addresses these limitations by:

- **Retrieval:**
    - When a user asks a question, the RAG system first retrieves relevant information from an external knowledge base (e.g., a database, document repository, or the internet).
    - This retrieval process often involves using techniques like vector databases to find information that is semantically like the user's query.
- **Augmentation:**
    - The retrieved information is then added to the user's prompt, providing the LLM with the necessary context to answer the question accurately.
- **Generation:**
    - The LLM uses both the original prompt and the retrieved information to generate a more informed and accurate response.

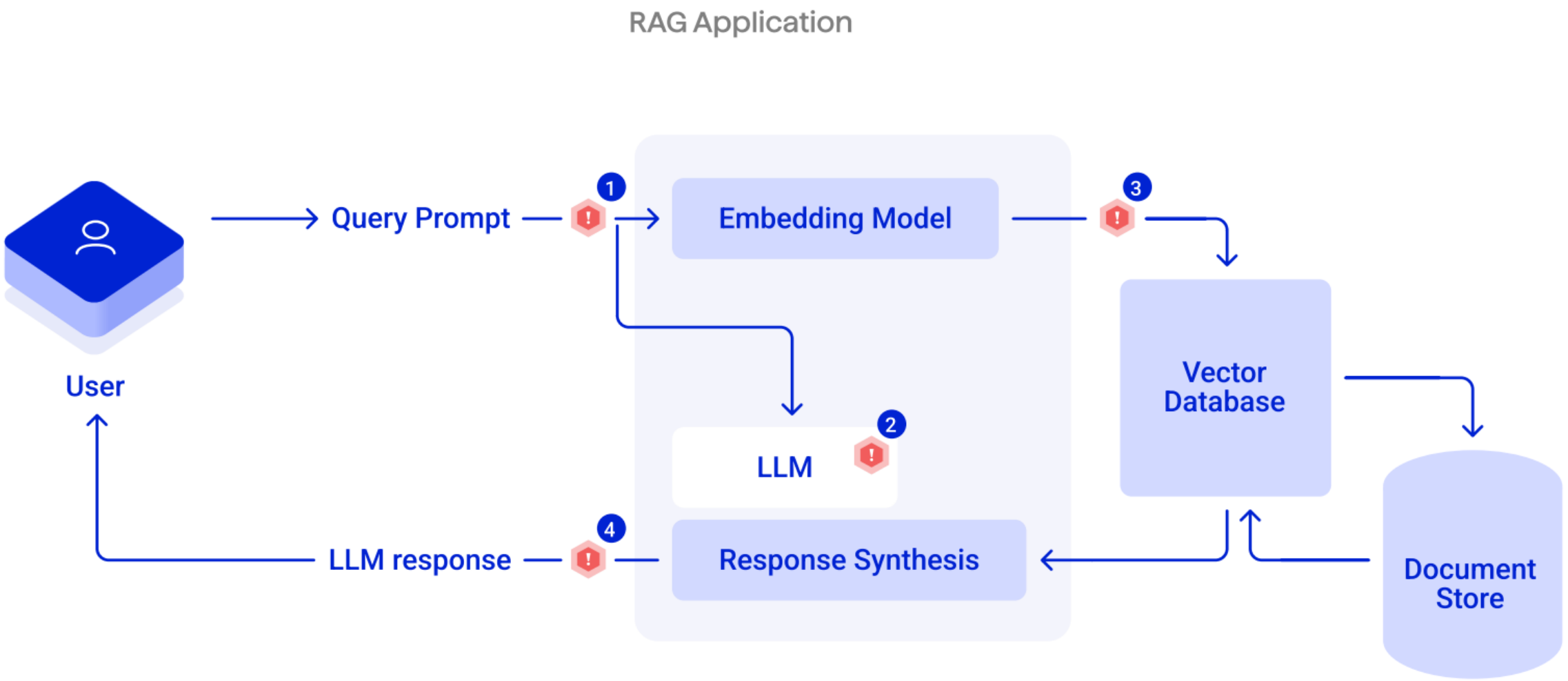
What is RAG – how does it help LLM

Key Benefits of RAG:

- **Improved Accuracy:** RAG reduces hallucinations by grounding the LLM's responses in factual information.
- **Access to Up-to-Date Information:** RAG allows LLMs to access and use real-time or frequently updated information.
- **Enhanced Domain Specificity:** RAG enables LLMs to provide accurate answers in specialized domains by retrieving information from relevant knowledge bases.
- **Increased Transparency:** RAG can provide citations or sources for the information used, making the LLM's responses more transparent and verifiable.

In essence, RAG bridges the gap between the LLM's inherent knowledge and the vast amount of information available externally, resulting in more reliable and informative AI interactions.

RAG Model Architecture with Threats



Harpreet Singh - Enterprise Security Architect

Common Threats to RAG Architecture

Common threats to RAG applications arise when there is

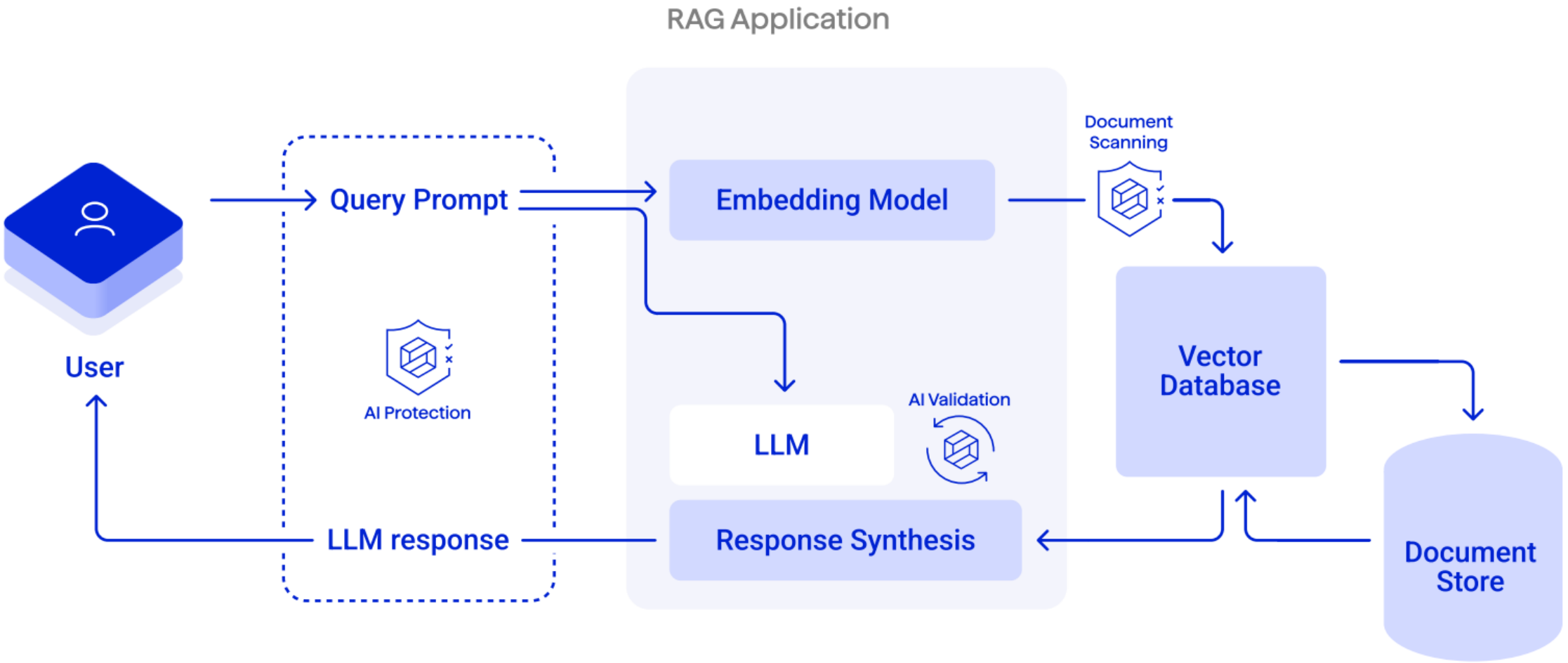
(1) **untrusted input** – input is not from a trusted source, may be poisoned or spoofed

(2) **misaligned model** (for example, through fine-tuning) - refers to a situation where the different components of the system—particularly the retrieval and generation phases—don't work together harmoniously, leading to inaccurate or unreliable outputs.

(3) **indirect prompt injection through untrusted documents** - a security vulnerability where malicious instructions are hidden within documents that the LLM is designed to process causing exploitation by attackers

(4) **unvalidated output** – refers to the LLM's generated response that hasn't been checked or verified against a reliable source or set of criteria. This poses a significant risk because RAG systems, while aiming to improve accuracy, can still produce incorrect or misleading information

Non-Threat based RAG Model Architecture



Harpreet Singh - Enterprise Security Architect

Threats Mitigation for RAG Architecture

1. Before selecting and after fine-tuning the LLM - AI validation should be applied to check for alignment, safety, and security.

AI Alignment - means that the model's responses are not only accurate but also ethical, unbiased, and helpful

AI Safety - focuses on preventing AI systems from causing unintended harm, also includes mitigating risks like the generation of harmful or misleading content.

- **Hallucinations:** RAG models can still generate false or misleading information, even when retrieving context. Ensuring factual consistency is paramount.
- **Data contamination:** If the retrieved data contains harmful or biased information, the model may perpetuate those issues.
- **Output control:** Ensuring that the model's output is appropriate and avoids generating toxic or dangerous content.

AI Security - involves protecting AI systems from malicious attacks and vulnerabilities, safeguarding the model and its underlying data from unauthorized access and manipulation

- **Prompt injection:** Attackers may attempt to manipulate the model's behavior by injecting malicious prompts.
- **Data leaks:** RAG systems that access sensitive data must implement robust access controls and encryption to prevent leaks.
- **Data poisoning:** Attackers could attempt to corrupt the retrieved data, leading to the generation of harmful responses.
- **Access control:** It is very important to make sure that only authorized users have access to certain datasets, and that the RAG model respects those access controls.

Harpreet Singh - Enterprise Security Architect

Threats Mitigation for RAG Architecture

2. At runtime, requests and responses should be subject to real-time AI protection to prevent attempts to misuse/abuse the application and to flag unsafe output.

Core Aspects of AI Protection in RAG:

- **Data Security:**
  - This involves protecting the data sources that the RAG model accesses.
  - Key measures include:
    - **Encryption:** Encrypting data at rest and in transit to prevent unauthorized access.
    - **Access Control:** Implementing strict access controls to ensure only authorized users and systems can access sensitive data.
    - **Data Anonymization:** Removing or masking sensitive information from data before it's used by the RAG model.
    - **Data Isolation:** Separating sensitive data into isolated environments.
- **Prompt Security:**
  - Protecting the RAG model from malicious prompts that could manipulate its behaviour.
  - Key measures include:
    - **Input Validation:** Thoroughly validating and sanitizing user inputs to prevent prompt injection attacks.
    - **Prompt Engineering Guardrails:** Implementing prompt engineering techniques that set boundaries and constraints on the model's output.
    - **Query validation:** Scrutinizing user queries before processing them.

Threats Mitigation for RAG Architecture

- **Output Security:**
  - Ensuring that the RAG model's output is safe, accurate, and appropriate.
  - Key measures include:
    - **Output Validation:** Implementing automated checks to verify the accuracy and relevance of the generated content.
    - **Content Filtering:** Filtering out harmful, toxic, or biased content from the model's output.
    - **Access control on output:** Limiting who has access to the generated results.
- **Model Security:**
  - Protecting the RAG model itself from unauthorized access and manipulation.
  - Key measures include:
    - **Secure Model Deployment:** Deploying the model in a secure environment with appropriate access controls.
    - **Model Monitoring:** Monitoring the model's behaviour for anomalous activity.
    - **Regular Security Audits:** Performing regular security assessments to identify and address vulnerabilities.

Threats Mitigation for RAG Architecture

- **Access Control:**
    - This is a large part of AI protection. Making sure that only authorized users, or systems can access the RAG system, and the data that it uses.
    - Implementing Role based access control, and other systems to limit access.

Why AI Protection is Crucial in RAG:

- RAG models often access sensitive data, so protecting that data is paramount.
- Malicious actors could exploit vulnerabilities in RAG systems to manipulate the model's behaviour or steal data.
- Ensuring the safety and reliability of RAG outputs is essential for building trust in AI systems.

By implementing robust AI protection measures, organizations can minimize the risks associated with RAG models and build secure and trustworthy AI applications

Harpreet Singh - Enterprise Security Architect

Threats Mitigation for RAG Architecture

3. Document scanning must be in place to prevent indirect prompt injection attempts – how does it work in RAG

Understanding the Threat: Indirect Prompt Injection

- **Direct Prompt Injection:**
  - This is where an attacker directly manipulates the user's input prompt to trick the RAG model. For example, they might insert malicious instructions into the query itself.
- **Indirect Prompt Injection:**
  - This is more insidious. Here, an attacker doesn't directly manipulate the user's immediate prompt. Instead, they inject malicious instructions into the *documents* that the RAG model retrieves.
  - When the RAG system retrieves these compromised documents as context, the model can be tricked into executing the attacker's instructions, even if the user's original query was harmless.

How Document Scanning Prevents This:

- **Proactive Security:**
  - Document scanning involves analysing the content of documents before they are indexed and made available to the RAG system.
- **Malicious Content Detection:**
  - The scanning process aims to identify and flag or remove potentially malicious content, such as:
    - Hidden instructions or commands.
    - Code snippets designed to exploit vulnerabilities.
    - Data that could manipulate the model's output in undesirable ways.

Threats Mitigation for RAG Architecture

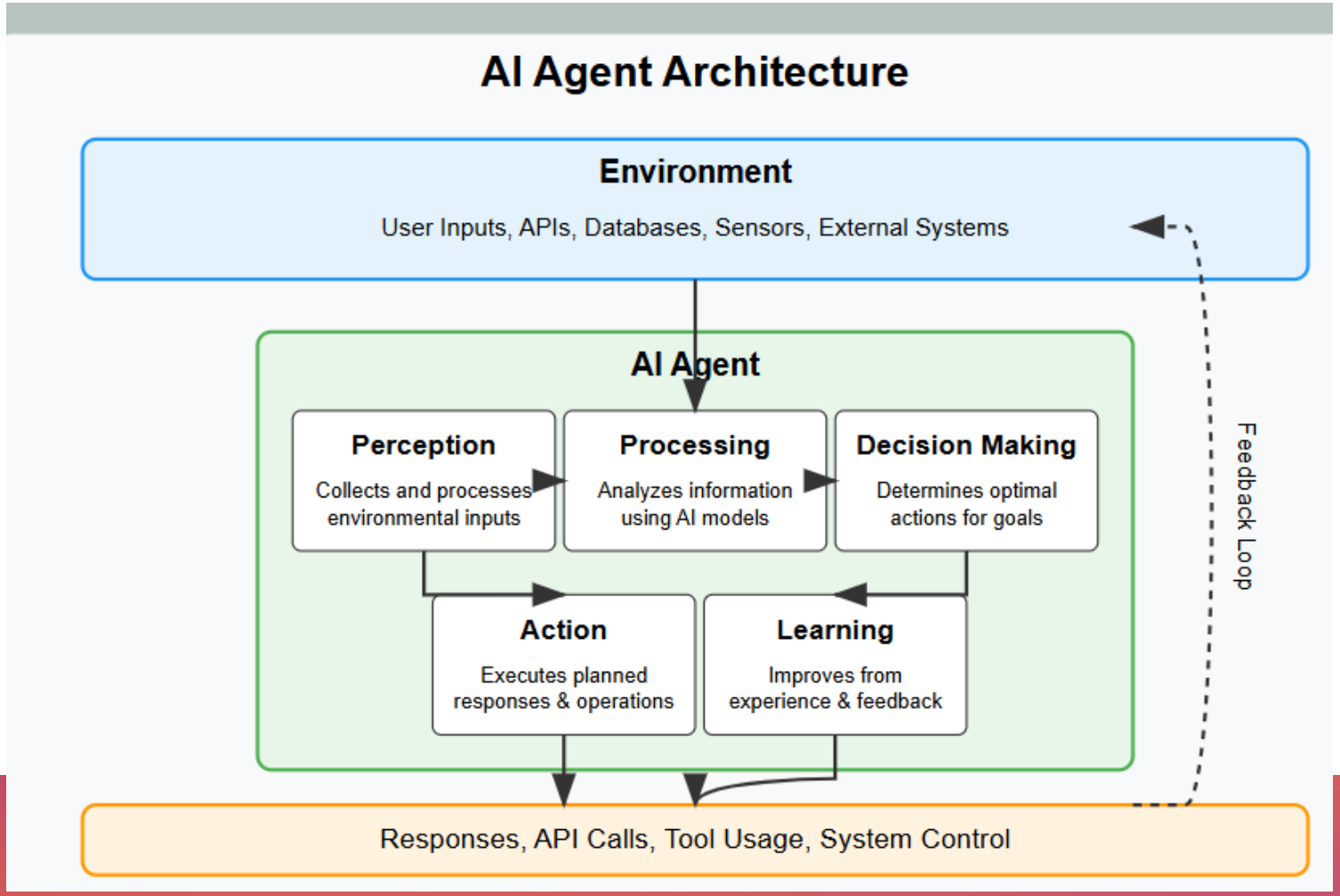
- **Content Sanitization:**
    - Beyond simply flagging malicious content, document scanning can also involve sanitizing documents by removing or neutralizing potentially harmful elements.
- **Maintaining Data Integrity:**
    - By scanning documents, you are helping to make sure that the data that the RAG system uses is clean, and has not been tampered with.

Why This is Crucial for RAG:

- **Dependency on External Data:**
    - RAG models rely on external data sources, which may be beyond the direct control of the system's developers.
- **Potential for Compromised Data:**
    - If an attacker can compromise these data sources, they can effectively control the RAG model's behaviour.
- **Mitigation of Hidden Threats:**
    - Document scanning is a defensive measure that helps to protect against threats that are not immediately obvious.

Harpreet Singh - Enterprise Security Architect

What is an AI Agent - How does it work



An AI agent is a software program designed to interact with its environment, perceive the data it receives, and take actions based on that data to achieve specific goals. AI agents simulate intelligent behavior, and they can be as simple as rule-based systems or as complex as advanced machine learning models. They use predetermined rules or trained models to make decisions and might need external control or supervision

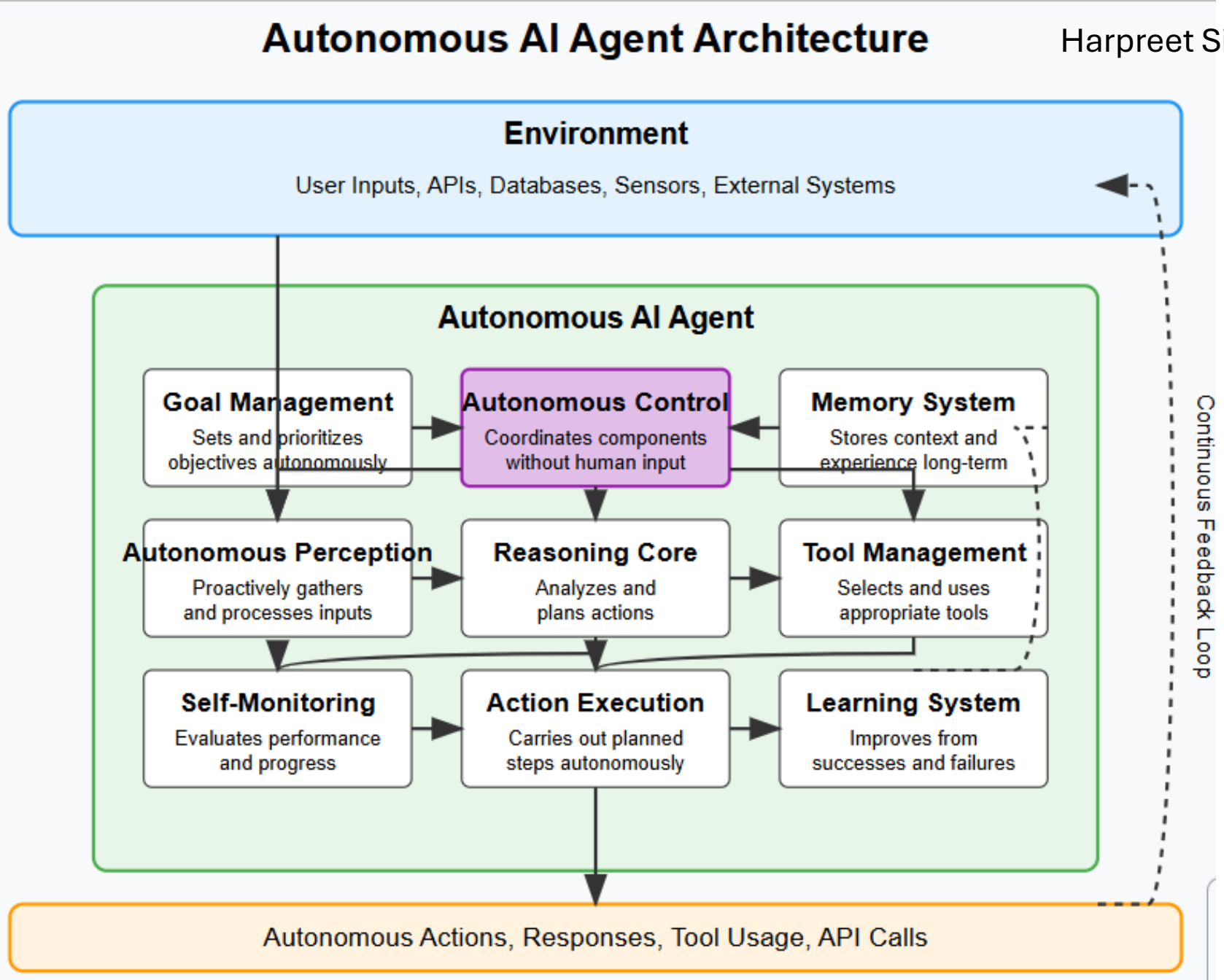
Summary - How does it work

This diagram illustrates the five key components of an AI agent:

- **Perception**: The agent collects information from its environment, which includes user inputs, APIs, databases, sensors, and other external systems.
- **Processing**: The agent analyses this information using its underlying AI models, such as large language models or other machine learning systems.
- **Decision Making**: Based on its analysis, the agent determines what actions to take to achieve its goals.
- **Action**: The agent executes its chosen actions, which might include generating responses, making API calls, using tools, or controlling systems.
- **Learning and Adaptation**: The agent improves over time based on experience and feedback.

The diagram also shows the important feedback loop, where the outputs of the agent influence future inputs, creating a continuous cycle of interaction and improvement. This architecture enables AI agents to operate autonomously and adapt to changing conditions while pursuing their designated objectives.

Harpreet Singh - Enterprise Security Architect



What is an Autonomous AI Agent, How does it work

- **An autonomous AI agent is an advanced software program that can operate independently without human control.**
- **It can think, act, and learn on its own, without needing constant input from humans.**
- **These agents are widely used in different industries, like healthcare, finance, and banking, to make things run smoother and more efficiently. They can adjust to new situations, learn from their experiences, and make decisions using their own internal systems**

Summary - How does it work

Harpreet Singh - Enterprise Security Architect

Autonomous AI agents are more advanced than standard AI systems because they can operate independently to achieve goals with minimal human supervision. These agents combine perception, reasoning, planning, and action to navigate their environments and complete tasks on their own.

Here's how autonomous AI agents work:

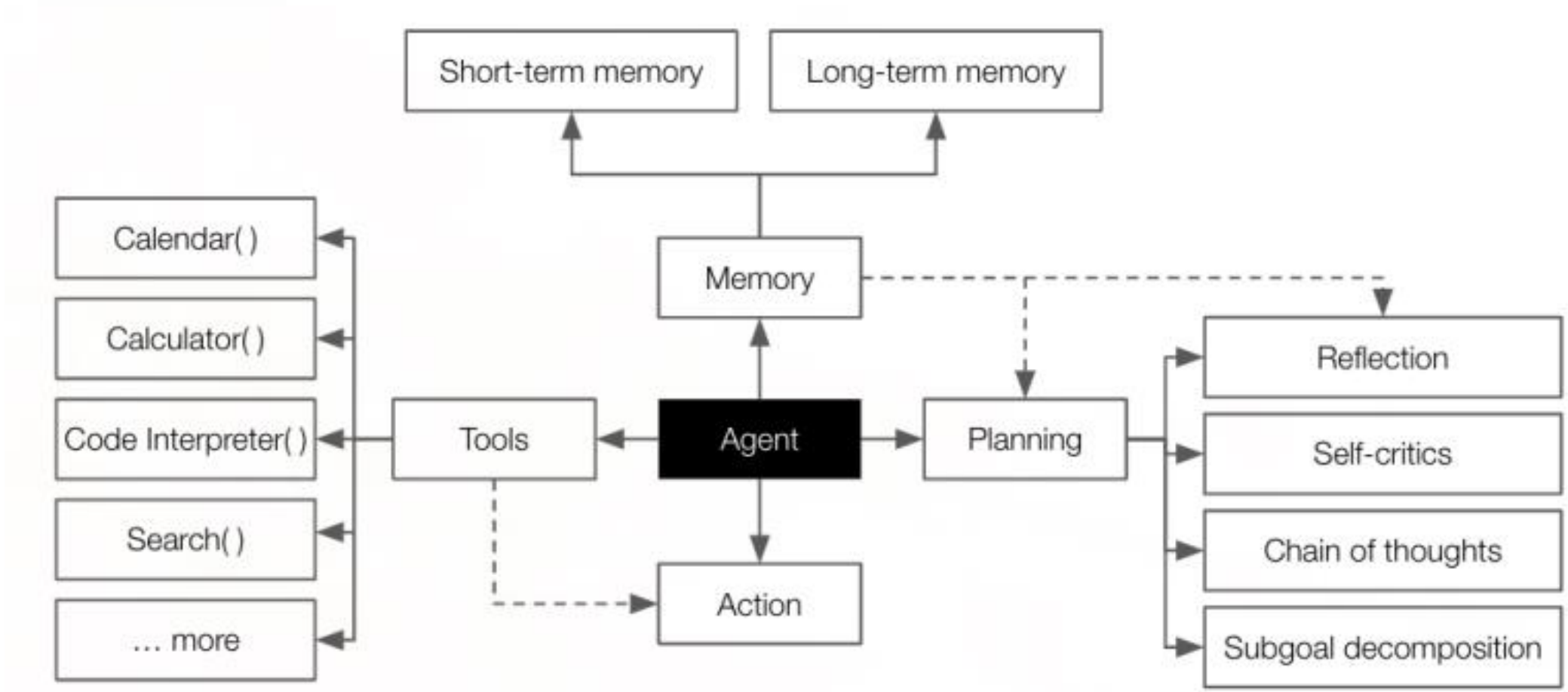
1. **Goal Setting**: The agent begins with defined objectives or can determine its own goals based on context
2. **Autonomous Perception**: Continuously collects and processes environmental data without human prompting
3. **Planning & Reasoning**: Creates multi-step plans to achieve goals, evaluating different approaches
4. **Tool Selection**: Independently chooses and uses appropriate tools and APIs from its available toolkit
5. **Execution & Monitoring**: Carries out actions while tracking progress and results
6. **Self-Correction**: Detects when plans aren't working and adapts strategies accordingly
7. **Memory & Learning**: Maintains context across sessions and improves from experience

Key Differences of Autonomous AI Agents

What makes autonomous AI agents truly autonomous is their ability to:

1. **Self-direct**: They make decisions without requiring human input for each step
2. **Chain actions**: They can execute complex sequences of operations across multiple systems
3. **Handle failures**: When one approach doesn't work, they can try alternative methods
4. **Maintain persistent state**: They keep track of progress across multiple interactions
5. **Balance exploration and exploitation**: They can decide when to gather more information versus when to act

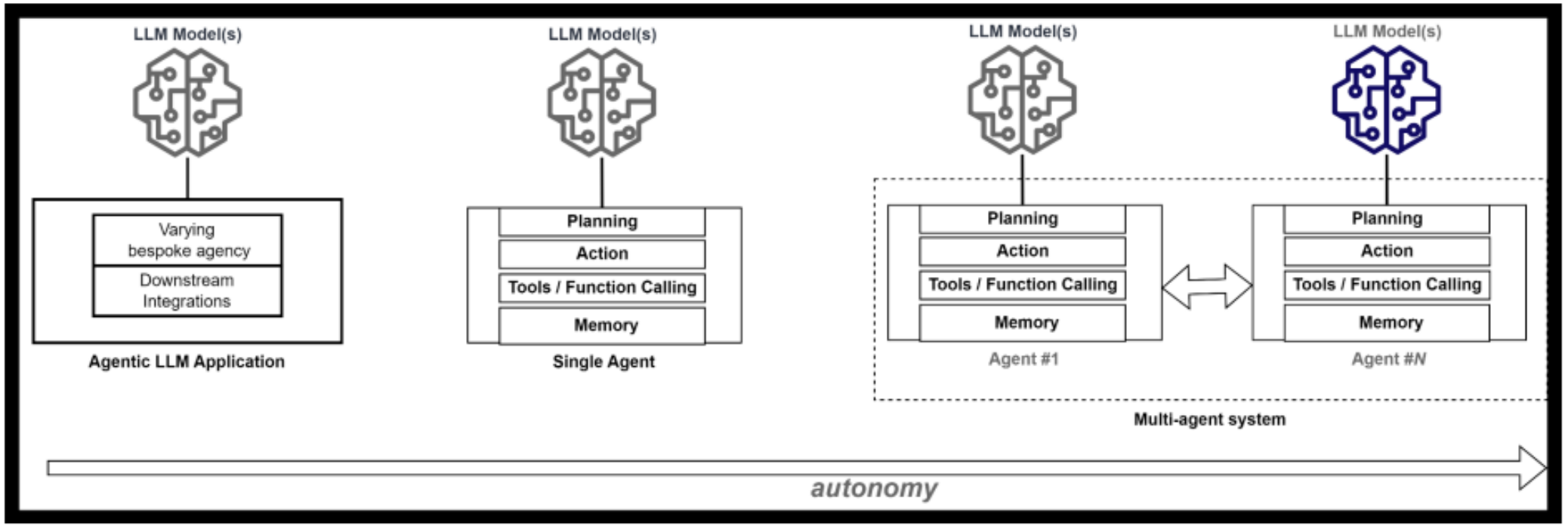
Flowchart or Architectural Diagram depicting an agent-based cognitive system



Harpreet Singh - Enterprise Security Architect

Agents & LLM Applications

LLM applications can exhibit agency and agentic behavior as described in the OWASP Top 10 for LLM Applications as part of the Excessive Agency and agents can be written as a LLM applications with the ability to reason and act using tools like APIs, databases and so on beyond than just generating text-based output



Increasingly, developers use agentic AI frameworks, which encapsulate agentic capabilities and offer greater productivity and reuse. Popular frameworks include **LangChain/LangFlow**, **AutoGen**, **CrewAI**, and so on.

Harpreet Singh - Enterprise Security Architect

Agentic AI Threat Model - OWASP

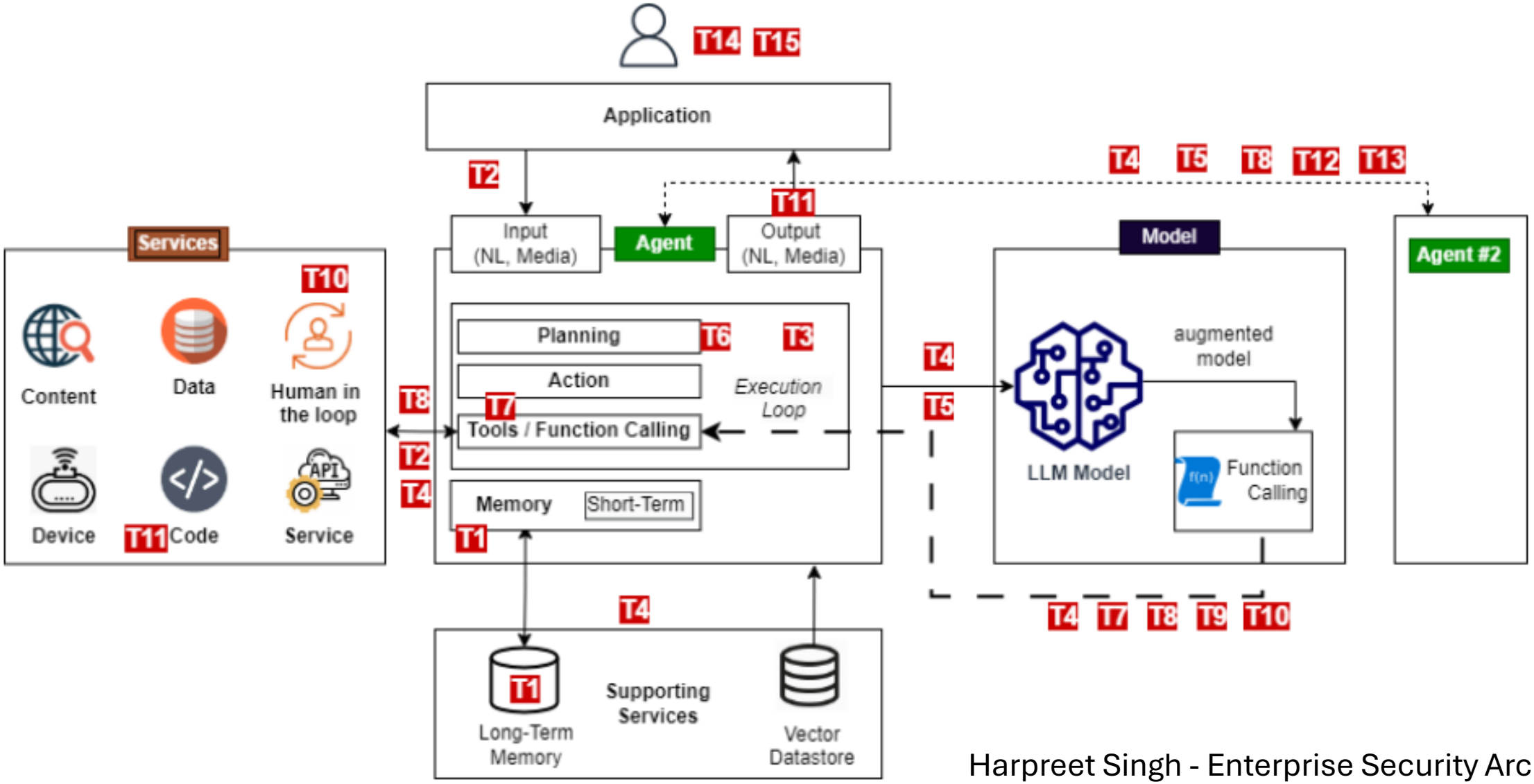
Threat modeling approach Threat modeling is a structured, repeatable process for identifying and mitigating security risks in a system. It involves analyzing a system from an adversarial perspective, identifying potential threats, and determining appropriate defenses. Ideally integrated into the software development lifecycle (SDLC), threat modeling is an ongoing process that evolves with the system. As outlined in the Threat Modeling Manifesto, it addresses four key questions:

- What are we working on?
- What can go wrong?
- What are we going to do about it?
- Did we do a good enough job?

There are established methodologies, such as STRIDE or PASTA that help practitioners perform threat modeling, but they are rooted in traditional cyber vulnerabilities and must be expanded or mapped to AI vulnerabilities.
You can find our more about threat modelling in application development and threat modelling methodologies in https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html
The GenAI Red Teaming guide from the OWASP Top 10 for LLM Project discusses Threat Modeling for Generative AI/LLM Systems https://genai.owasp.org/resource/genai-red-teaming-guide/

Harpreet Singh - Enterprise Security Architect

Agent AI based Threat Model – OWASP (OWASP Top 10 for LLM Apps & Gen AI Agentic Security Initiative) – **Feb 2025**



Harpreet Singh - Enterprise Security Architect

Agent AI based Threat Model – OWASP (OWASP Top 10 for LLM Apps & Gen AI Agentic Security Initiative) – **Feb 2025**

Detailed Threat Model:

TID	Threat Name	Threat Description	Mitigations
T1	**Memory Poisoning**	Memory Poisoning involves exploiting an AI's memory systems, both short and long-term, to introduce malicious or false data and exploit the agent's context. This can lead to altered decision-making and unauthorized operations.	Implement memory content validation, session isolation, robust authentication mechanisms for memory access, anomaly detection systems, and regular memory sanitization routines. Require AI-generated memory snapshots for forensic analysis and rollback if anomalies are detected.
T2	**Tool Misuse**	Tool Misuse occurs when attackers manipulate AI agents to abuse their integrated tools through deceptive prompts or commands, operating within authorized permissions. This includes **Agent Hijacking**, where an AI agent ingests adversarial manipulated data and subsequently executes unintended actions, potentially triggering malicious tool interactions. For more information on Agent Hijacking see https://www.nist.gov/news-	Enforce strict tool access verification, monitor tool usage patterns, validate agent instructions, and set clear operational boundaries to detect and prevent misuse. Implement execution logs that track AI tool calls for anomaly detection and post-incident review.

		events/news/2025/01/technical-blog-strengthening-ai-agent-hijacking-evaluations	
T3	**Privilege Compromise**	Privilege Compromise arises when attackers exploit weaknesses in permission management to perform unauthorized actions. This often involves dynamic role inheritance or misconfigurations.	Implement granular permission controls, dynamic access validation, robust monitoring of role changes, and thorough auditing of elevated privilege operations. Prevent cross-agent privilege delegation unless explicitly authorized through predefined workflows.
T4	**Resource Overload**	Resource Overload targets the computational, memory, and service capacities of AI systems to degrade performance or cause failures, exploiting their resource-intensive nature.	Deploy resource management controls, implement adaptive scaling mechanisms, establish quotas, and monitor system load in real-time to detect and mitigate overload attempts. Implement AI rate-limiting policies to restrict high-frequency task requests per agent session.
T5	**Cascading Hallucination Attacks**	These attacks exploit an AI's tendency to generate contextually plausible but false information, which can propagate through systems and disrupt decision-making. This can also lead to destructive reasoning affecting tools invocation.	Establish robust output validation mechanisms, implement behavioral constraints, deploy multi-source validation, and ensure ongoing system corrections through feedback loops. Require secondary validation of AI-generated knowledge before it is used in critical decision-making processes. This will face the same constraints of scaling AI as discussed in Overwhelming Human In the Loop and would require similar approaches.

Agent AI based Threat Model - OWASP

T6	**Intent Breaking & Goal Manipulation**	This threat exploits vulnerabilities in an AI agent's planning and goal-setting capabilities, allowing attackers to manipulate or redirect the agent's objectives and reasoning. One common approach is Agent Hijacking mentioned in Tool Misuse.	Implement planning validation frameworks, boundary management for reflection processes, and dynamic protection mechanisms for goal alignment. Deploy AI behavioral auditing by having another model check the agent and flag significant goal deviations that could indicate manipulation.
T7	**Misaligned & Deceptive Behaviors**	AI agents executing harmful or disallowed actions by exploiting reasoning and deceptive responses to meet their objectives.	Train models to recognize and refuse harmful tasks, enforce policy restrictions, require human confirmations for high-risk actions, implement logging and monitoring. Utilize deception detection strategies such as behavioral consistency analysis, truthfulness verification models, and adversarial red teaming to assess inconsistencies between AI outputs and expected reasoning pathways. This threat at an early stage but both Anthropic and OpenAI have published some work in this area (see https://www.anthropic.com/research/towards-

			understanding-sycophancy-in-language-models and https://openai.com/index/faulty-reward-functions/)
T8	**Repudiation & Untraceability**	Occurs when actions performed by AI agents cannot be traced back or accounted for due to insufficient logging or transparency in decision-making processes.	Implement comprehensive logging, cryptographic verification, enriched metadata, and real-time monitoring to ensure accountability and traceability. Require AI-generated logs to be cryptographically signed and immutable for regulatory compliance.
T9	**Identity Spoofing & Impersonation**	Attackers exploit authentication mechanisms to impersonate AI agents or human users, enabling them to execute unauthorized actions under false identities.	Develop comprehensive identity validation frameworks, enforce trust boundaries, and deploy continuous monitoring to detect impersonation attempts. Use behavioral profiling, involving a second model, to detect deviations in AI agent activity that may indicate identity spoofing.
T10	**Overwhelming Human in the Loop**	This threat targets systems with human oversight and decision validation, aiming to exploit human cognitive limitations or compromise interaction frameworks.	Develop advanced human-AI interaction frameworks, and adaptive trust mechanisms. These are dynamic AI governance models that employ dynamic intervention thresholds to adjust the level of human oversight and automation based on risk, confidence, and context. Apply hierarchical AI-human collaboration where low-risk decisions are automated, and human intervention is prioritized for high-risk anomalies.
T11	**Unexpected RCE and Code Attacks**	Attackers exploit AI-generated execution environments to inject malicious code, trigger unintended system behaviors, or execute unauthorized scripts.	Restrict AI code generation permissions, sandbox execution, and monitor AI-generated scripts. Implement execution control policies that flag AI-generated code with elevated privileges for manual review.

T12	**Agent Communication Poisoning**	Attackers manipulate communication channels between AI agents to spread false information, disrupt workflows, or influence decision-making.	Deploy cryptographic message authentication, enforce communication validation policies, and monitor inter-agent interactions for anomalies. Require multi-agent consensus verification for mission-critical decision-making processes.
T13	**Rogue Agents in Multi-Agent Systems**	Malicious or compromised AI agents operate outside normal monitoring boundaries, executing unauthorized actions or exfiltrating data.	Restrict AI agent autonomy using policy constraints and continuous behavioral monitoring. While cryptographic attestation mechanisms for LLMs do not yet exist, agent integrity can be maintained via controlled hosting environments, regular AI red teaming, and input/output monitoring for deviations

T14	**Human Attacks on Multi-Agent Systems**	Adversaries exploit inter-agent delegation, trust relationships, and workflow dependencies to escalate privileges or manipulate AI-driven operations.	Restrict agent delegation mechanisms, enforce inter-agent authentication, and deploy behavioral monitoring to detect manipulation attempts. Enforce multi-agent task segmentation to prevent attackers from escalating privileges across interconnected agents.
T15	**Human Manipulation**	In scenarios where AI agents engage in direct interaction with human users, the trust relationship reduces user skepticism, increasing reliance on the agent's responses and autonomy. This implicit trust and direct human/agent interaction create risks, as attackers can coerce agents to manipulate users, spread misinformation, and take covert actions.	Monitor agent behavior to ensure it aligns with its defined role and expected actions. Restrict tool access to minimize the attack surface, limit the agent's ability to print links, implement validation mechanisms to detect and filter manipulated responses using guardrails, moderation APIs, or another model

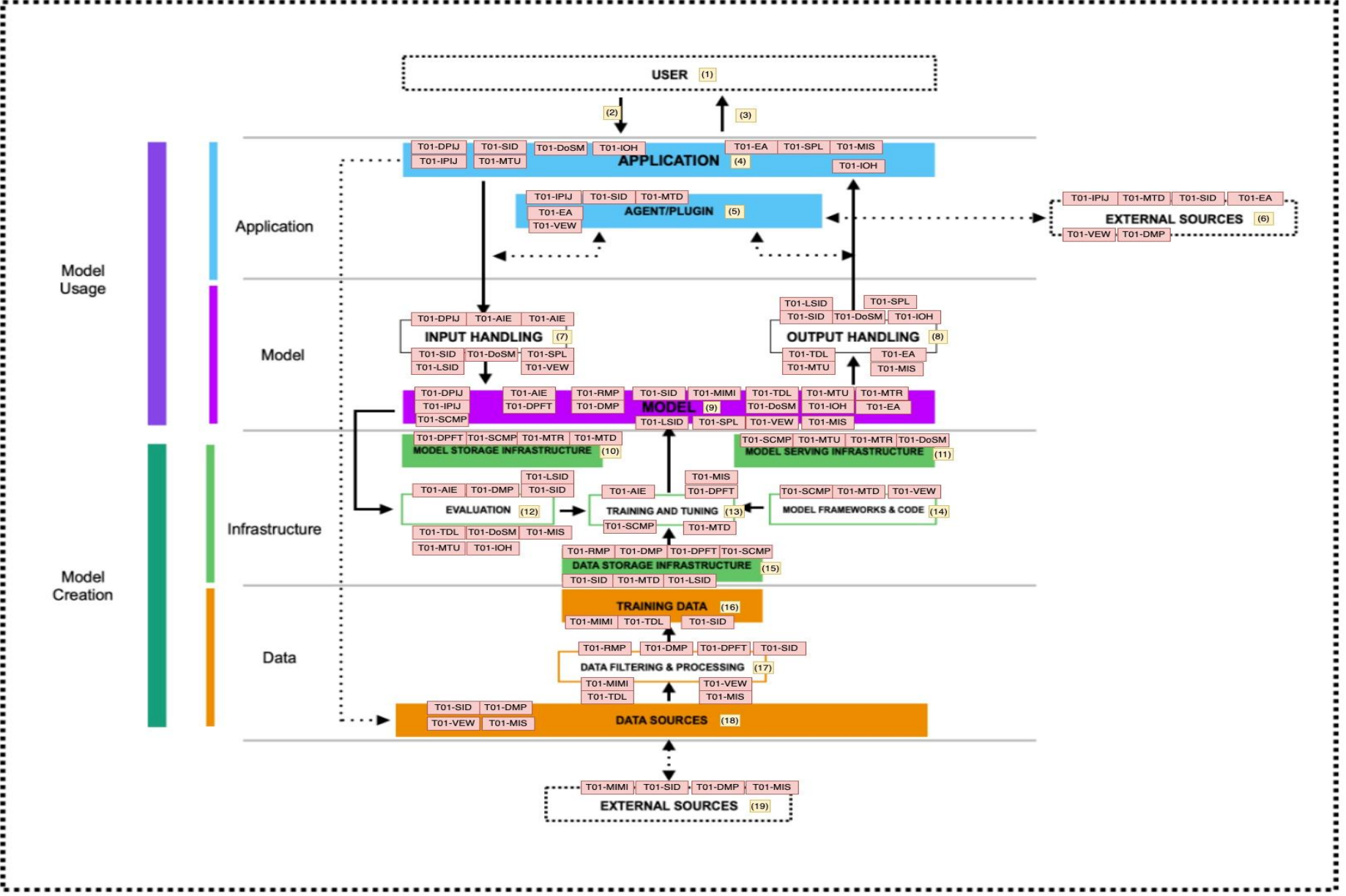
OWASP Top 10 for Large Language Models (LLM) and Generative AI Attacks

1. **Prompt Injection** - Attackers manipulate the model with crafted inputs that override instructions or intended constraints, causing the system to perform unintended actions.
2. **Insecure Output Handling** - Failing to properly validate, sanitize or encode AI-generated outputs, potentially leading to XSS, SSRF, or code injection vulnerabilities when outputs are used in applications.
3. **Training Data Poisoning** - Deliberately corrupting or manipulating training data to introduce vulnerabilities or biases that can be exploited later.
4. **Model Denial of Service** - Overwhelming the model with computationally expensive requests that exhaust computational resources or cause excessive costs.
5. **Supply Chain Vulnerabilities** - Security issues in pre-trained models, third-party datasets, or AI development frameworks that introduce inherited vulnerabilities.

OWASP Top 10 for Large Language Models (LLM) and Generative AI Attacks

6. **Sensitive Information Disclosure** - Models inadvertently revealing confidential information from training data or revealing proprietary prompt engineering techniques.
7. **Insecure Plugin Design** - Security flaws in LLM plugins that fail to properly validate inputs or handle permissions, allowing attackers to exploit connected systems.
8. **Excessive Agency** - Granting AI systems too much autonomy without appropriate oversight, enabling harmful autonomous actions or decisions.
9. **Overreliance** - Excessive trust in AI-generated content without verification, potentially leading to security issues, disinformation, or poor decisions.
10. **Model Theft** - Unauthorized access to proprietary models through extraction attacks or unauthorized access, leading to intellectual property loss or competitive disadvantage.

These vulnerabilities represent significant security challenges specific to LLM and generative AI implementations that organizations should address as part of their security strategy.

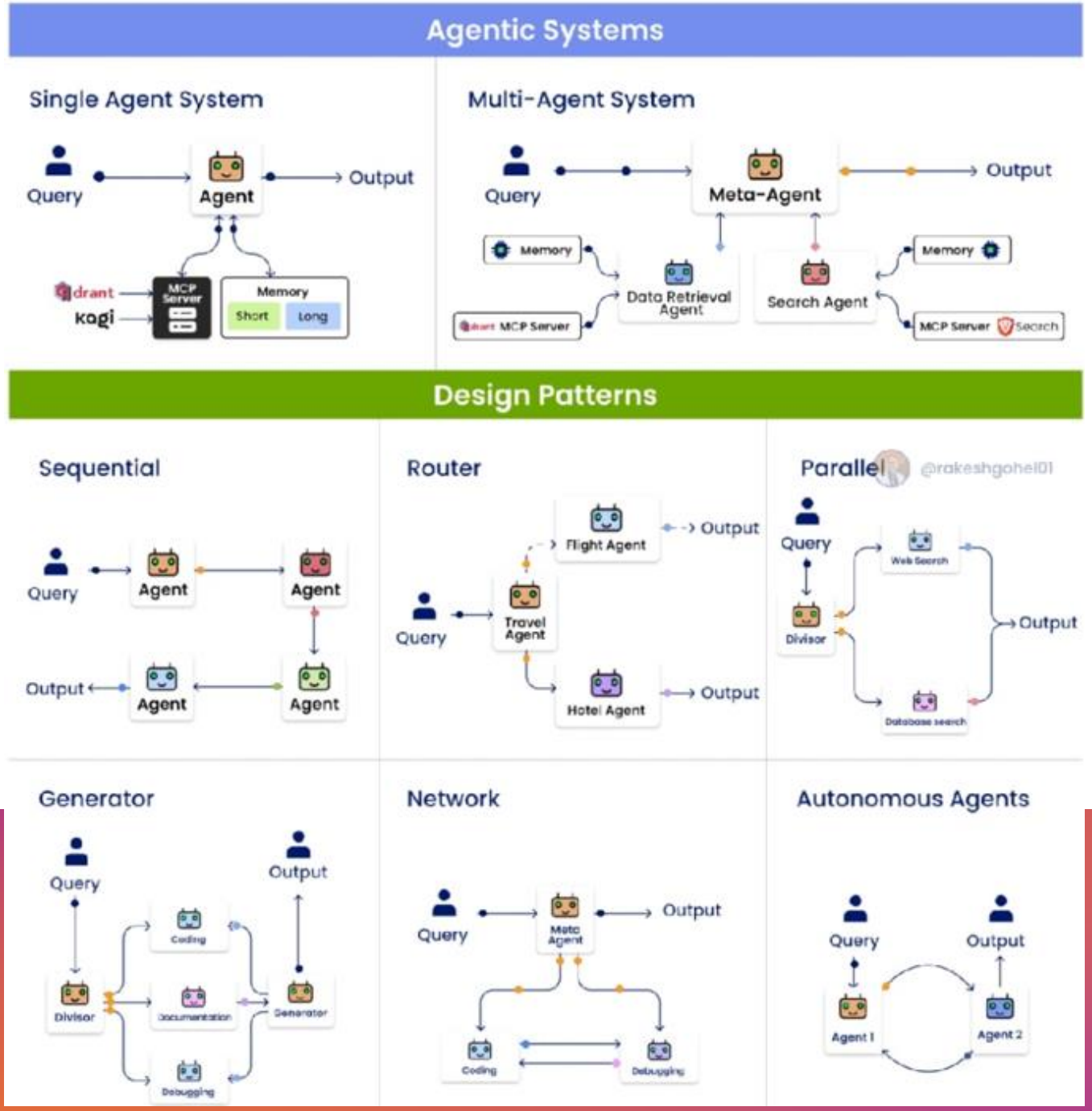


OWASP (LLM & AI Exchange) Threats	
Threat Model ID	**Threat OWASP ID: Description**
T01-DPIJ	LLM01: Direct Prompt Injection (DPIJ))
T01-IPIJ	LLM01:Indirect Prompt Injection (IPIJ)
T01-AIE	Threat 2.1: Adversarial Input Evasion (AIE)
T01-RMP	LLM04: RunTime Data and Model Poisoning (RMP)
T01-DMP	LLM04: Data Model Poisoning (DMP)
T01-DPFT	LLM04: Data and Model Poisoning Fine-tuning (DPFT)
T01-SCMP	LLM03 Supply Chain Model Poisoning (SCMP)
T01-SID	LLM02 Sensitive Information Disclosure (SID)
T01-MIMI	Threat 2.3.2: Model Inversion & Membership Inference (MIMI)
T01-TDL	Threat 3.2: Training Data Leakage (TDL)
T01-MTU	Threat 2.4: Model Theft Through Use (MTU)
T01-MTR	Threat 4.3: Model Theft at Runtime (MTR)
T01-MTD	Threat 3.2: Model Theft During Development (MTD)
T01-DoSM	LLM10: Denial of Service of Model Services (DoSM)
T01-LSID	LLM02: Leak Sensitive input Data (LSID)
T01-IOH	LLM05: Improper Output Handling (IOH)
T01-EA	LLM06: Excessive Agency (EA)
T01-SPL	LLM07: System Prompt Leakage (SPL)
T01-VEW	LLM08: View & Embedding Weakness (VEW)
T01-MIS	LLM09: Misinformation (MIS)

SAIF Components	
Name (SAIF ID)	**Description**
User (1)	Person or System Interacting with AI Application
User Input (2)	Inputs of Commands Submitted by User/System
User Output (3)	Outputs Returned by AI Application
Application (4)	Logic that Receives User I/O & Interacts with Model
Agents/Plugins (5)	Interfaces Extending App/Model (e.g APIs, RAG tools)
External Sources (6)	Databases, Services or APIs used by Agents/Plugins
Input Handling (7)	Input Validation and Sanitization
Output Handling (8)	Filters and Sanitizes Outputs Including Data Redaction
Model (9)	AI Model Code and Weights Created with Training Data
Model Storage Infra. (10)	Storage for Training & AI Model Data (separate)
Model Serving Infra. (11)	System/Processes to Deploy AI Model in Production
Evaluation (12)	Evaluate AI Model for Accuracy, Bias, Drift, Policies
Training & Tuning (13)	Manages AI. Model Training & Tuning from Data
Model Frmk. & Code (14)	Code & Frameworks Necessary to Train the AI Model
Data Storage Infra (15)	Storage for Training Data and the AI Model (separate)
Training Data (16)	Data to Train the AI Model Patterns and Make Predictions
Data Filtering & Processing (17)	Sanitizes, Validates and Transforms Data Before Training
Data Sources (18)	Data used for Model Training (e.g. API, DBs, sensors)
External Sources (19)	Third Party (External) Data Feeds For Model Training

OWASP Threat Model Diagram (LLM & AI Threats)

Harpreet Singh - Enterprise Security Architect



Agentic Systems Overview – Multi Agent Design Pattern

The diagram illustrates two main categories of agentic systems:

Single Agent System: A straightforward architecture where a single agent processes queries using memory components (short and long-term) and produces outputs.

Multi-Agent System: A more complex setup featuring a Meta-Agent that coordinates between specialized agents like Data Retrieval Agent and Search Agent, each with their own memory and MCP (Model Context Protocol) servers.

Harpreet Singh - Enterprise Security Architect

Six Key Design Patterns

1. Sequential Pattern

Agents work in a linear chain where each agent processes the output from the previous one before passing it to the next. This creates a pipeline-like workflow for complex tasks requiring multiple processing steps.

2. Router Pattern

A central routing mechanism (Flight Agent) directs queries to appropriate specialized agents (Travel Agent, Hotel Agent) based on the nature of the request. This enables intelligent task distribution.

3. Parallel Pattern

Multiple agents work simultaneously on different aspects of the same query, with their outputs being consolidated. This pattern enables concurrent processing for improved efficiency.

4. Generator Pattern

Features a cyclical workflow with specialized roles - a Generator creates content, a Discriminator evaluates it, and an Accumulator collects results. This pattern is useful for iterative content creation and refinement

5. Network Pattern

Agents are interconnected in a network topology where they can communicate with multiple other agents. A Meta Agent coordinates the network, with agents like Coding and Debugging working collaboratively.

6. Autonomous Agents Pattern

Independent agents (Agent 1, Agent 2) operate autonomously while maintaining communication channels. This pattern supports distributed decision-making and self-directed task execution.

These patterns provide different approaches to organizing AI agents based on factors like task complexity, coordination requirements, and processing efficiency needs

Building an AI CoE – High Performance Delivery

BUILDING BLOCKS OF AN AI COE

Innovation

Establish strong AI foundation with a culture of AI innovation and thought leadership.

Talent

Acquire and retain top AI talent. Foster collaboration with key business stakeholders and global teams on AI use-cases.

Governance

Develop AI strategy aligned with organizational goals. Integrate Trusted AI principles into the AI lifecycle to manage Risk.

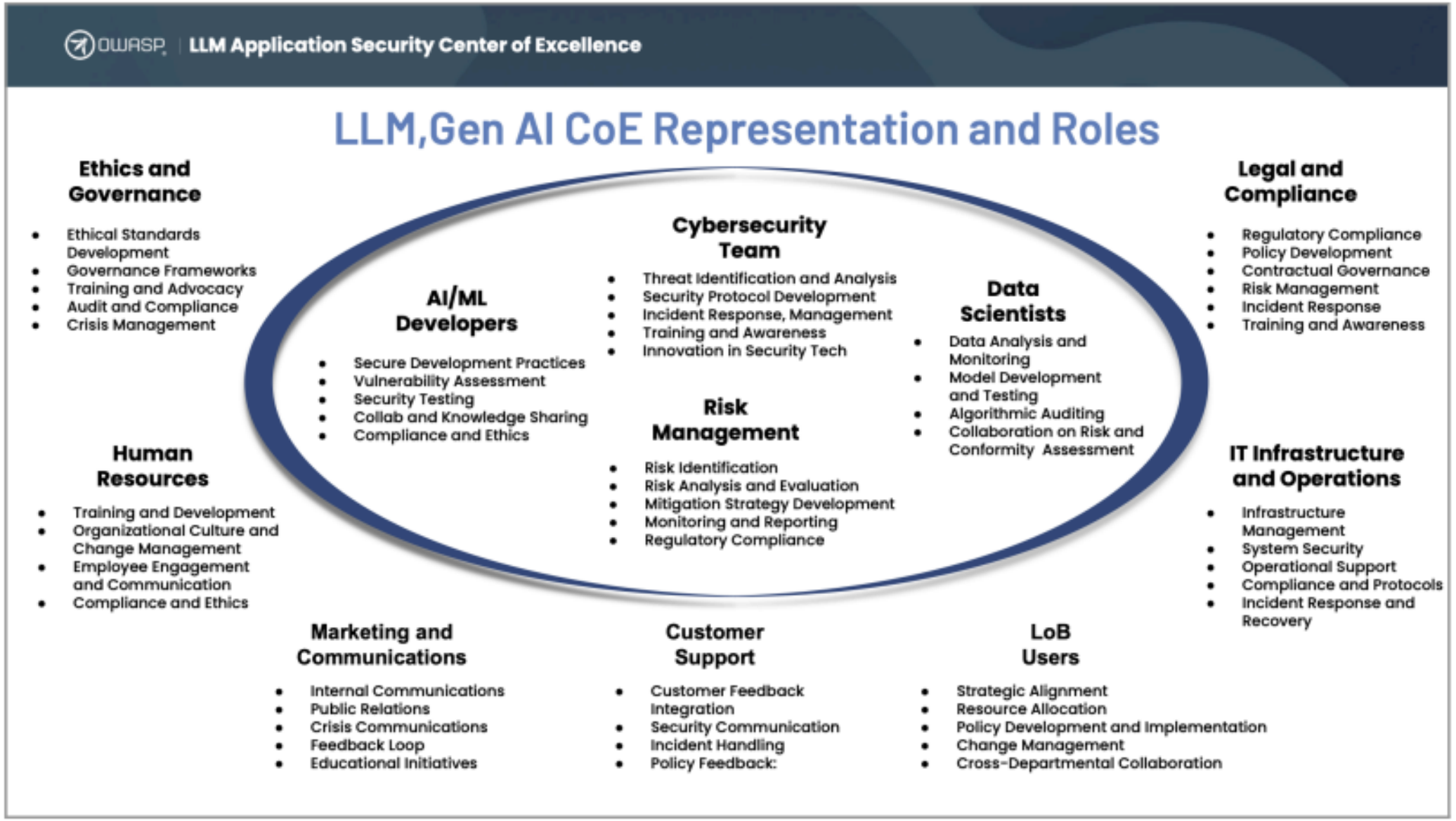
Data Readiness

Improve Enterprise Data Quality and focus on Unstructured data to maximize the potential of AI.

Enablement

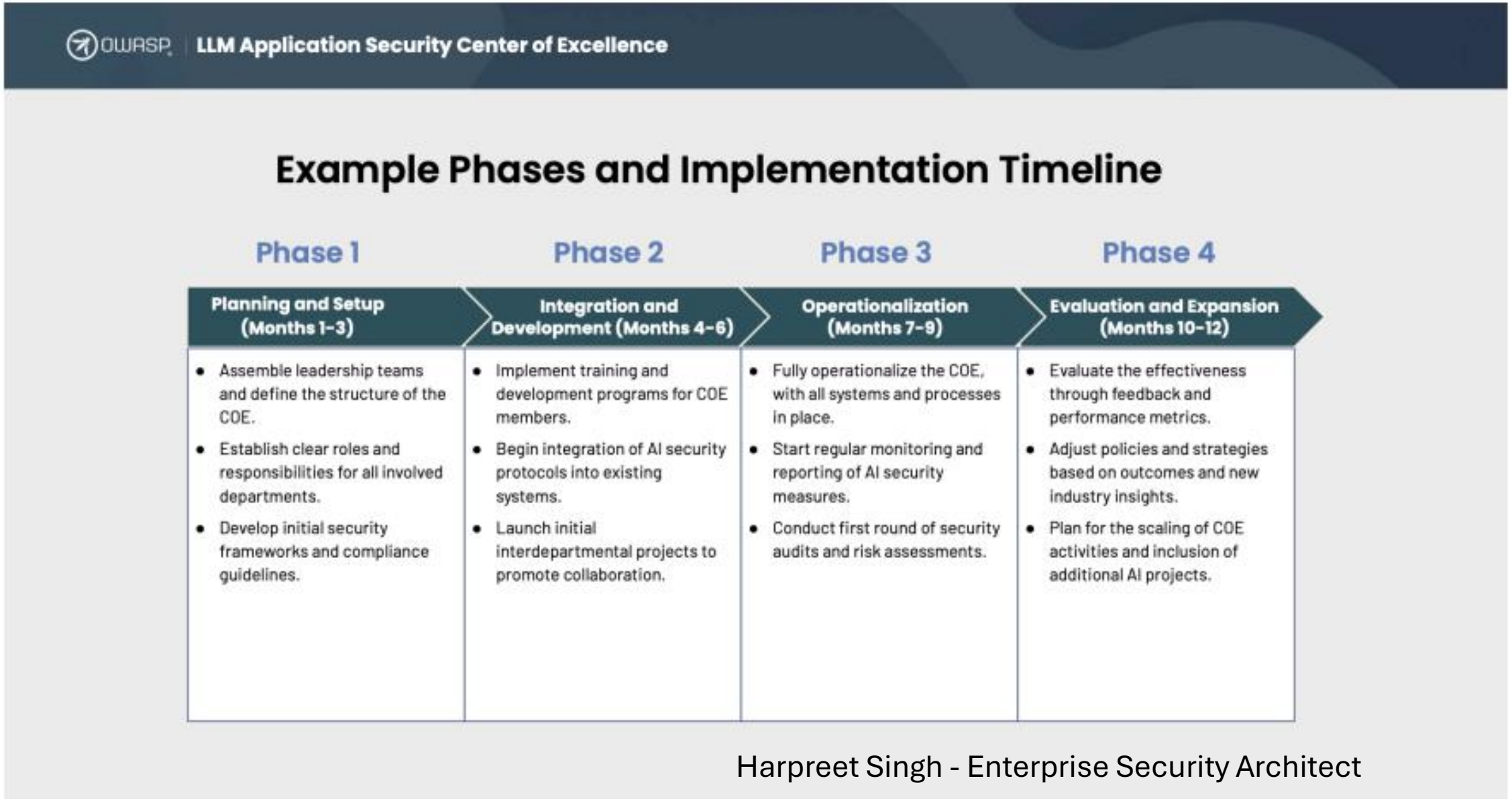
Drive AI-first mindset at by AI literacy, training, knowledge-sharing and change management.

OWASP's recommendation on LLM, GEN AI CoE



Harpreet Singh - Enterprise Security Architect

OWASP's recommendation on LLM, GEN AI CoE



Harpreet Singh - Enterprise Security Architect

OWASP's recommendation on LLM, GEN AI CoE

Phase 1: Planning and Setup (Months 1-3)

During the initial phase, the focus is on establishing the foundational structure of the COE. This involves assembling a leadership team from across key departments, defining the specific roles and responsibilities within the COE, and developing initial security frameworks and policies. The objective is to create a robust organizational structure that supports effective communication and collaboration across diverse teams, setting the stage for the subsequent integration and development phases.

Key Actions (Phase 1)

- Assemble leadership teams and define the structure of the COE.
- Establish clear roles and responsibilities for all involved departments.
- Develop initial security frameworks and compliance guidelines.

Phase 2: Integration and Development (Months 4-6)

This phase is critical for integrating AI security protocols into existing systems and fostering interdepartmental collaboration. The COE will implement comprehensive training programs to ensure all members are up-to-date with the latest security practices and technologies. Additionally, the launch of initial interdepartmental projects aims to enhance collaborative efforts and begin the practical application of the developed security frameworks, thereby testing and refining these strategies in real-world scenarios.

Key Actions (Phase 2)

- Implement training and development programs for COE members.
- Begin integration of AI security protocols into existing systems.
- Launch initial interdepartmental projects to promote collaboration.

Phase 3: Operationalization (Months 7-9)

The COE becomes fully functional in the operationalization phase with all systems and processes actively running. Regular monitoring and reporting mechanisms are established to track the effectiveness of AI security measures. Security audits and risk assessments are conducted to identify any vulnerabilities and to ensure compliance with the established security protocols. This phase is crucial for adjusting operational workflows based on feedback and ensuring that the COE's activities are effectively enhancing AI security.

Key Actions (Phase 3)

- Fully operationalize the COE, with all systems and processes in place.
- Start regular monitoring and reporting of AI security measures.
- Conduct first round of security audits and risk assessments.

Phase 4: Evaluation and Expansion (Months 10-12)

The final phase focuses on evaluating the COE's effectiveness through comprehensive reviews and feedback mechanisms. Based on the outcomes of these evaluations, adjustments are made to policies and strategies. Additionally, plans for scaling COE activities are developed to include additional AI projects and initiatives, aiming to broaden the scope and impact of the COE's work. This phase ensures that the COE remains dynamic and adaptable to new challenges and opportunities.

Key Actions (Phase 4)

- Evaluate the effectiveness through feedback and performance metrics.
- Adjust policies and strategies based on outcomes and new industry insights.
- Plan for the scaling of COE activities and inclusion of additional AI projects.

Emerging Trend in AI Security

Emerging Trends in AI Security

As AI technologies rapidly evolve, so do the security challenges and solutions. Key emerging trends include:

1. **AI-powered cybersecurity**: Using AI to detect and respond to threats more quickly and effectively than traditional methods.
2. **Adversarial AI**: The rise of AI systems designed to attack other AI systems, necessitating robust defenses.
3. **Federated learning**: Enhancing privacy by training AI models on distributed datasets without centralizing the data.
4. **Quantum-resistant cryptography**: Preparing for the potential threat quantum computing poses to current encryption methods.
5. **Ethical AI regulations**: Increasing focus on regulatory frameworks to ensure responsible AI development and deployment.
6. **AI supply chain security**: Growing emphasis on securing the entire AI development pipeline, from data collection to model deployment.
7. **Explainable AI (XAI)**: Developing AI systems that can provide clear explanations for their decisions, crucial for security auditing and trust-building.

CoEs must stay abreast of these trends to effectively anticipate and address evolving security challenges in the AI landscape.

Harpreet Singh - Enterprise Security Architect

OWASP's Top 10 on LLM with reference(s)

OWASP Top 10 for Large Language Models (LLM)

The OWASP Top 10 for LLM applications identifies the most critical security vulnerabilities specifically targeting large language model systems. Here's a detailed breakdown:

1. Prompt Injection

Security issues where attackers craft inputs that manipulate the model to:

- Override system instructions/guardrails
- Extract confidential prompt information
- Make the model perform unintended actions
- Execute "jailbreaks" that circumvent security controls

2. Insecure Output Handling

Failure to properly validate or sanitize AI-generated outputs, leading to:

- Cross-site scripting (XSS)
- SQL injection when outputs are used in databases
- Server-side request forgery (SSRF)
- Code injection vulnerabilities

Harpreet Singh - Enterprise Security Architect

OWASP's Top 10 on LLM with reference(s)

3. Training Data Poisoning

Manipulation of training datasets to compromise model behaviour:

- Deliberate introduction of biases or backdoors
- Insertion of harmful content that can later be triggered
- Targeting model behaviour to produce exploitable outputs

4. Model Denial of Service

Attacks that aim to overwhelm or degrade LLM service:

- Computationally expensive prompts that consume excessive resources
- Crafted inputs that cause abnormally long processing times
- Requests that trigger maximum token usage or API costs

5. Supply Chain Vulnerabilities

Security weaknesses inherited through the AI development ecosystem:

- Pre-trained models with embedded vulnerabilities
- Compromised third-party datasets
- Malicious or vulnerable AI frameworks and libraries

6. Sensitive Information Disclosure

When models inadvertently reveal information they shouldn't:

- Training data leakage (exposing private data used in training)
- System prompt/instruction leakage
- Proprietary algorithm details or implementation specifics

7. Insecure Plugin Design

Security flaws in LLM plugin ecosystems:

- Insufficient validation of plugin inputs
- Improper permission controls for plugin actions
- Vulnerable integrations with external systems

8. Excessive Agency

Risks from granting AI systems too much autonomy:

- Unintended actions when connected to external systems
- Lack of human oversight for critical operations
- Insufficient control mechanisms for autonomous behaviour

Harpreet Singh - Enterprise Security Architect

9. Overreliance

Excessive trust in AI outputs without verification:

- Using AI-generated content without human validation
- Failing to implement proper fact-checking mechanisms
- Missing safeguards against hallucinations or fabrications

10. Model Theft

Unauthorized access to proprietary model assets:

- Model extraction through systematic querying
- Theft of model weights or architecture details
- Exploitation of API vulnerabilities to extract model capabilities

Potential attacks - Top 10 for Agentic AI

While OWASP hasn't released a formal "Top 10 for Agentic AI" specifically, we can identify the most critical security concerns for AI agents based on emerging threats and extensions of the LLM security framework:

1. Agent Prompt Injection

Attacks where malicious inputs manipulate an AI agent to perform unauthorized actions or bypass restrictions, potentially more severe than regular prompt injections as agents can take actions in connected systems.

2. Objective Manipulation

Attacks that subtly alter or redirect an agent's goals/objectives, causing it to optimize for unintended or harmful outcomes while appearing to function normally.

3. Tool/API Exploitation

Manipulating an agent to misuse its connected tools or APIs, potentially escalating privileges, accessing unauthorized systems, or executing harmful commands.

4. Confused Deputy Attacks

Tricking an agent into performing privileged operations on behalf of an attacker by exploiting the trust relationship between the agent and connected systems.

5. Planning Poisoning

Corrupting the agent's planning process to insert malicious steps that appear legitimate but serve an attacker's purposes.

6. Memory/Context Manipulation

Attacks targeting the agent's memory systems to insert false information, corrupt decision context, or exploit how agents store and retrieve information.

Harpreet Singh - Enterprise Security Architect

Potential attacks - Top 10 for Agentic AI

While OWASP hasn't released a formal "Top 10 for Agentic AI" specifically, we can identify the most critical security concerns for AI agents based on emerging threats and extensions of the LLM security framework:

7. Multi-Step Attack Chains

Complex attacks leveraging multiple vulnerabilities across the agent system, where early steps may appear benign but enable more serious later exploitation.

8. Autonomous Action Abuse

Exploiting the agent's autonomous capabilities to perform actions at scale or in ways that evade normal detection methods.

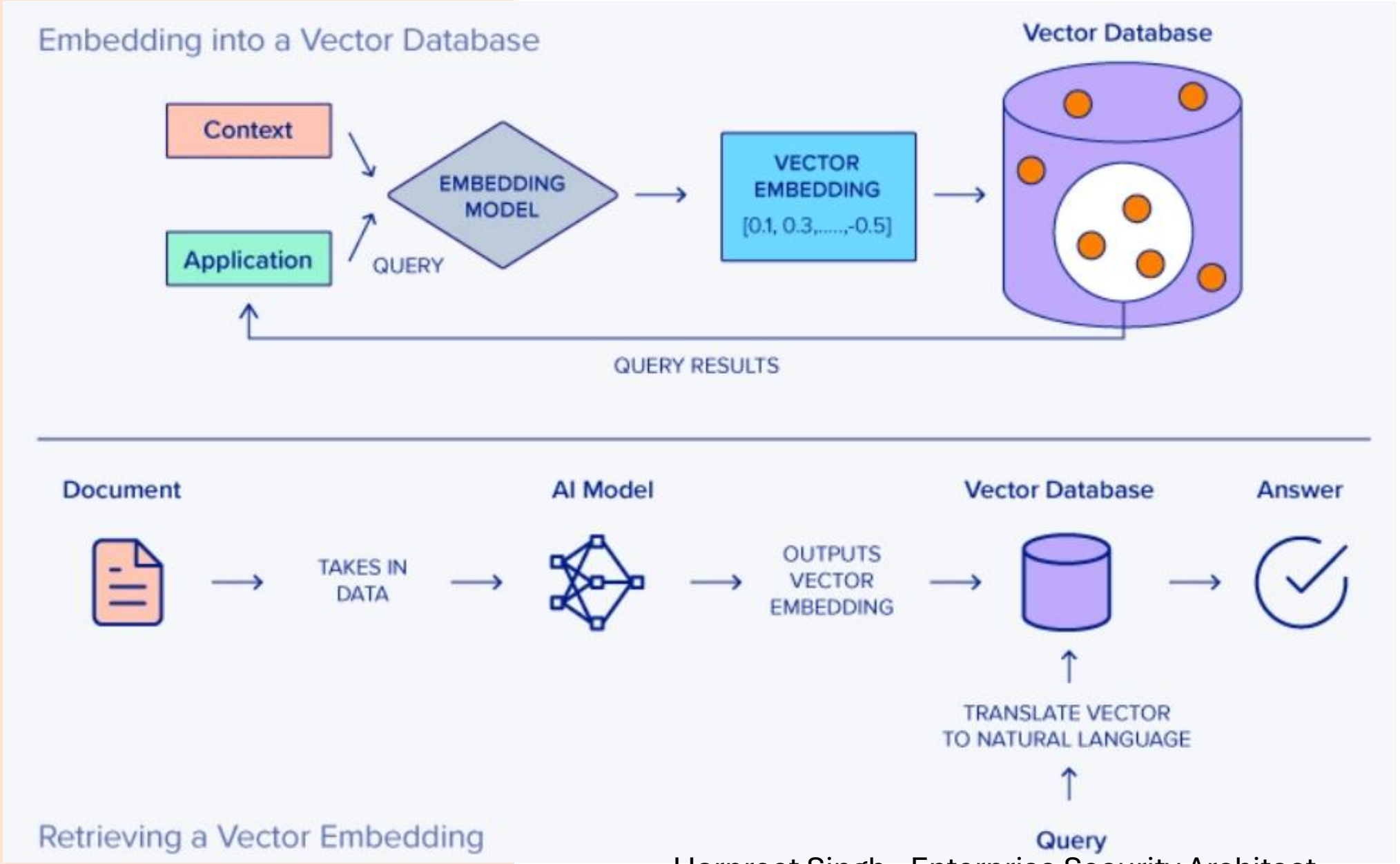
9. Decision Boundary Exploitation

Identifying and targeting edge cases in the agent's decision-making process to force unsafe or incorrect actions in boundary conditions.

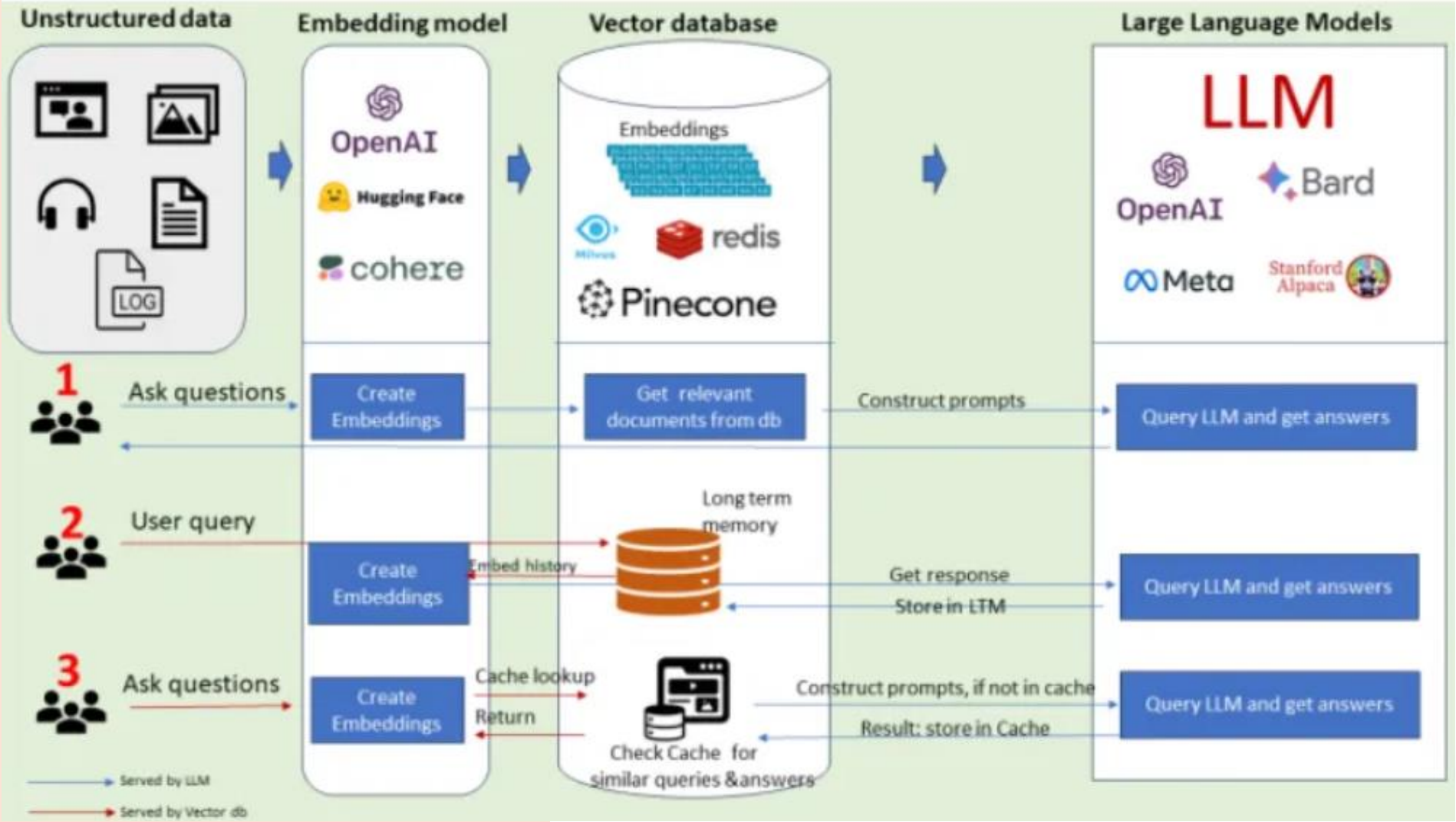
10. Feedback Loop Manipulation

Corrupting the agent's learning or adaptation mechanisms to progressively degrade security posture or create backdoors through poisoned feedback.

The Vector Database – What is it > Role of a Vec DB > How does it work



The Vector Database – What is it > Role of a Vec DB > How does it work



A Vector Database in the context of Generative AI (GenAI) and Large Language Models (LLMs) is a specialized type of database designed to efficiently store, manage, and retrieve high-dimensional vector embeddings. These embeddings are numerical representations of data, such as text, images, audio, or video, that capture the semantic meaning and relationships within the data

The Vector Database – The Vital role it plays

1. Understanding Semantic Meaning:

- LLMs excel at understanding and generating human language, but they require a way to process the meaning of words and concepts. Vector embeddings transform textual data into numerical vectors where words or phrases with similar meanings are located close to each other in a high-dimensional space.
- Vector databases store these embeddings, allowing for semantic search, where you can find information based on meaning rather than exact keywords. For example, searching for "big cat" could retrieve documents containing "lion," "tiger," or "panther."

2. Enhancing LLM Capabilities:

- **Retrieval-Augmented Generation (RAG):** Vector databases are fundamental in RAG systems. When a user asks a question, the system converts the query into a vector embedding and searches the vector database for semantically similar documents or chunks of information. This retrieved context is then added to the prompt given to the LLM, enabling it to generate more accurate, relevant, and fact-based answers grounded in external knowledge.
- **Overcoming Knowledge Cut-offs and Hallucinations:** LLMs have a limited knowledge base based on their training data, which has a cut-off date. By using a vector database with up-to-date information, LLMs can access and incorporate current knowledge into their responses, reducing the likelihood of generating incorrect or nonsensical information (hallucinations).
- **Providing Context and Memory:** LLMs are typically stateless, meaning they don't inherently remember past interactions. Vector databases can store embeddings of previous turns in a conversation, allowing the LLM to retrieve relevant context and maintain a more coherent and engaging dialogue.
- **Personalization and Recommendations:** By embedding user preferences and item features as vectors, vector databases can power recommendation systems that suggest similar content, products, or services based on semantic similarity.

Harpreet Singh - Enterprise Security Architect

The Vector Database – The Vital role it plays

3. Efficient Storage and Retrieval of Complex Data:

- GenAI often deals with vast amounts of unstructured data like text documents, images, and videos. Vector databases are optimized for storing and indexing these complex data types in their vector form.
- They employ specialized indexing techniques and similarity search algorithms (like nearest neighbour search) to quickly and efficiently retrieve relevant information from these high-dimensional vector spaces, even with billions of data points. This is much faster and more effective than traditional keyword-based searches on unstructured data.

4. Key Use Cases:

- **Question Answering Systems:** Building intelligent chatbots and Q&A systems that can understand the meaning of questions and retrieve relevant answers from a knowledge base.
- **Semantic Search:** Implementing search functionalities that go beyond keyword matching to understand the intent and context of user queries.
- **Recommendation Systems:** Creating personalized recommendations for products, movies, music, articles, and more.
- **Content-Based Retrieval:** Finding similar images, videos, or documents based on their content rather than metadata.
- **Anomaly Detection:** Identifying unusual patterns or outliers in data by comparing their vector embeddings to normal patterns.
- **Semantic Caching:** Storing embeddings of frequently asked questions and their answers to speed up response times and reduce computational costs.

In summary, a vector database acts as a crucial memory and knowledge retrieval system for GenAI and LLMs. By efficiently storing and searching through semantic representations of data, it enables these models to access relevant information, understand context, generate more accurate responses, and power a wide range of intelligent applications.

Harpreet Singh - Enterprise Security Architect

The Vector Database – How does it work

How does Vector Databases Work?

To understand the value of vector databases, we must first explore how they differ from traditional databases. Traditional databases store data in tables of rows and columns. Vector databases, in contrast, store data encoded as multi-dimensional numeric vectors.

Vector Representations

At the core of vector databases is the idea of representing data as numeric vectors. For example, an image of a cat may be encoded as a 512-dimensional vector like:

[0.23, 0.54, 0.32, ..., 0.12, 0.45, 0.90]

Text too can be represented as vectors derived from the words and semantics. Vectors serve as numeric digital signatures capturing the essence of data.

Generating Vectors

Vectors can be generated for data in various ways:

- **Machine Learning Models** — Models like Word2Vec, BERT, and CLIP create vectors reflecting data semantics.
- **Data Hashing** — Methods like **SimHash** and **MinHash** convert data to consistent vector hashes.
- **Data Indexing** — Text and image features are extracted, scaled and combined into vectors.

Vectorizing diverse data allows storing it in a consistent, unified way.

Harpreet Singh - Enterprise Security Architect

The Vector Database – How does it work

Storing Vectors Efficiently

Once data is vectorized, the vectors are persisted by the database. Key capabilities include:

- **Compact Storage** — Vectors compressed to optimize storage capacity.
- **Memory Caching** — Vectors cached in memory for faster retrieval.
- **Distributed Architecture** — Vectors sharded across nodes to scale capacity.
- **Columnar Data Layout** — Column store for efficient analytic querying.

By leveraging these techniques, vector databases can store vast vector data for AI applications.

Harpreet Singh - Enterprise Security Architect

The Vector Database – How does it work (application in LLM & GEN AI – further explained)

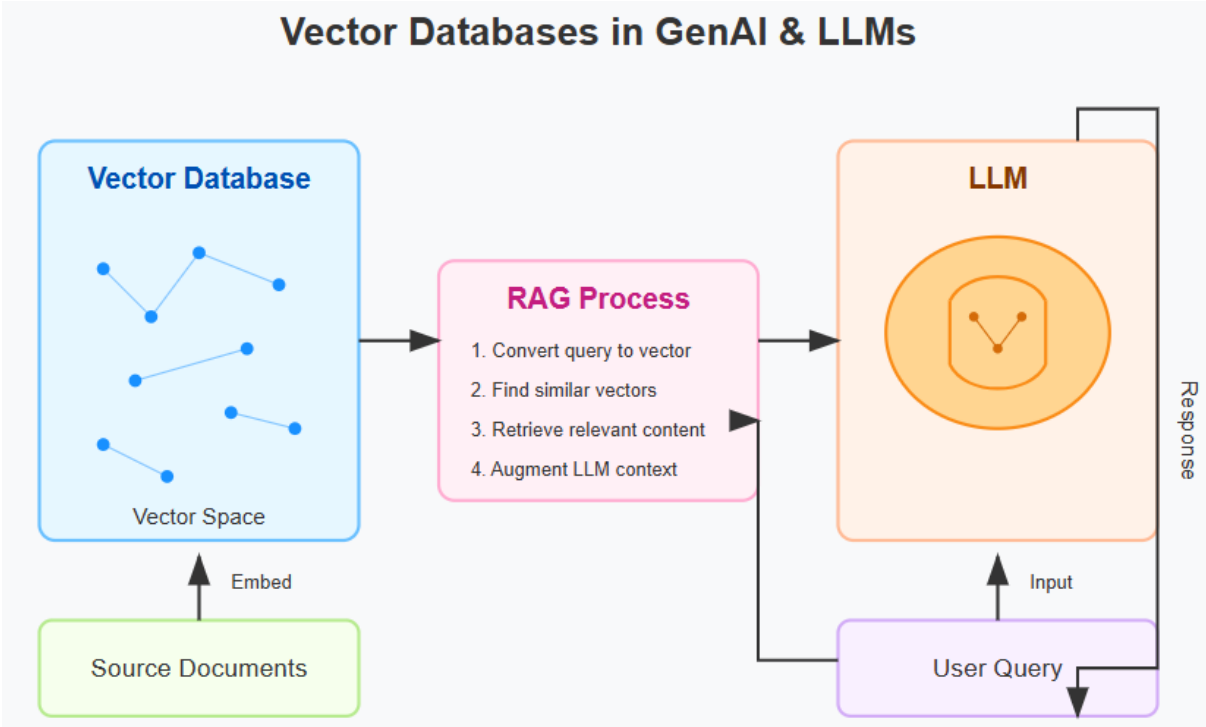
How Vector Databases Work with LLMs and GenAI

1. **Data Representation**: Raw data (text, images, etc.) is converted into numerical vector representations using embedding models
2. **Storage and Indexing**: These vectors are stored and indexed for efficient similarity searching
3. **Retrieval**: When queried, the database finds semantically similar vectors
4. **Integration with LLMs**: Retrieved information provides context to LLMs for more accurate responses

Key Workflows

LLM Application with RAG (Retrieval-Augmented Generation):

- User queries are converted to vector embeddings
- Similar vectors are retrieved from the database
- These relevant documents augment the LLM's context
- The LLM generates a response using both its training and the retrieved context



The Vector Database – How does it work (application in LLM & GEN AI – further explained)

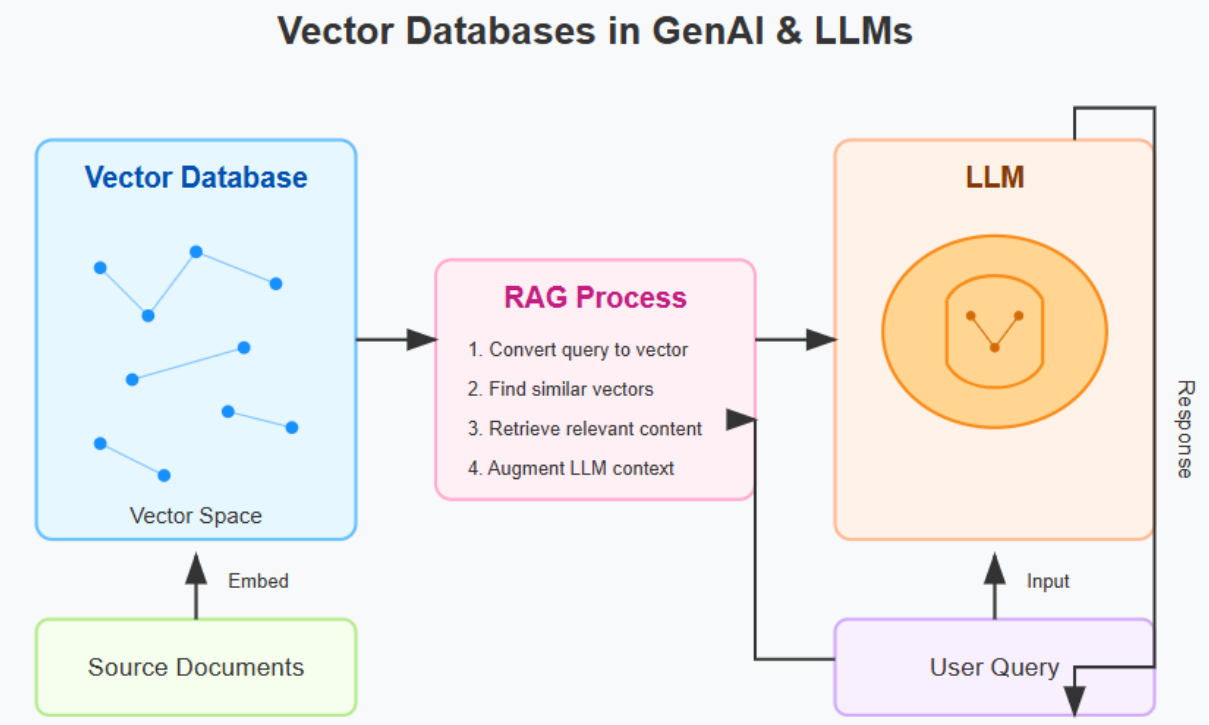
Core Concepts of Vector Databases

1. **Vector Embeddings**:
    - Numerical representations that capture semantic meaning
    - Typically high-dimensional vectors (e.g., 768, 1024, or 4096 dimensions)
    - Similar concepts have similar vector representations
2. **Similarity Search**:
    - Uses distance metrics like cosine similarity or Euclidean distance
    - Finds vectors closest to the query vector
    - More efficient than traditional keyword search for semantic matching
3. **Approximate Nearest Neighbor (ANN) Algorithms**:
    - Makes searching fast in high-dimensional spaces
    - Common techniques: HNSW (Hierarchical Navigable Small World), IVF (Inverted File Index), PQ (Product Quantization)

The Vector Database – How does it work (application in LLM & GEN AI – further explained)

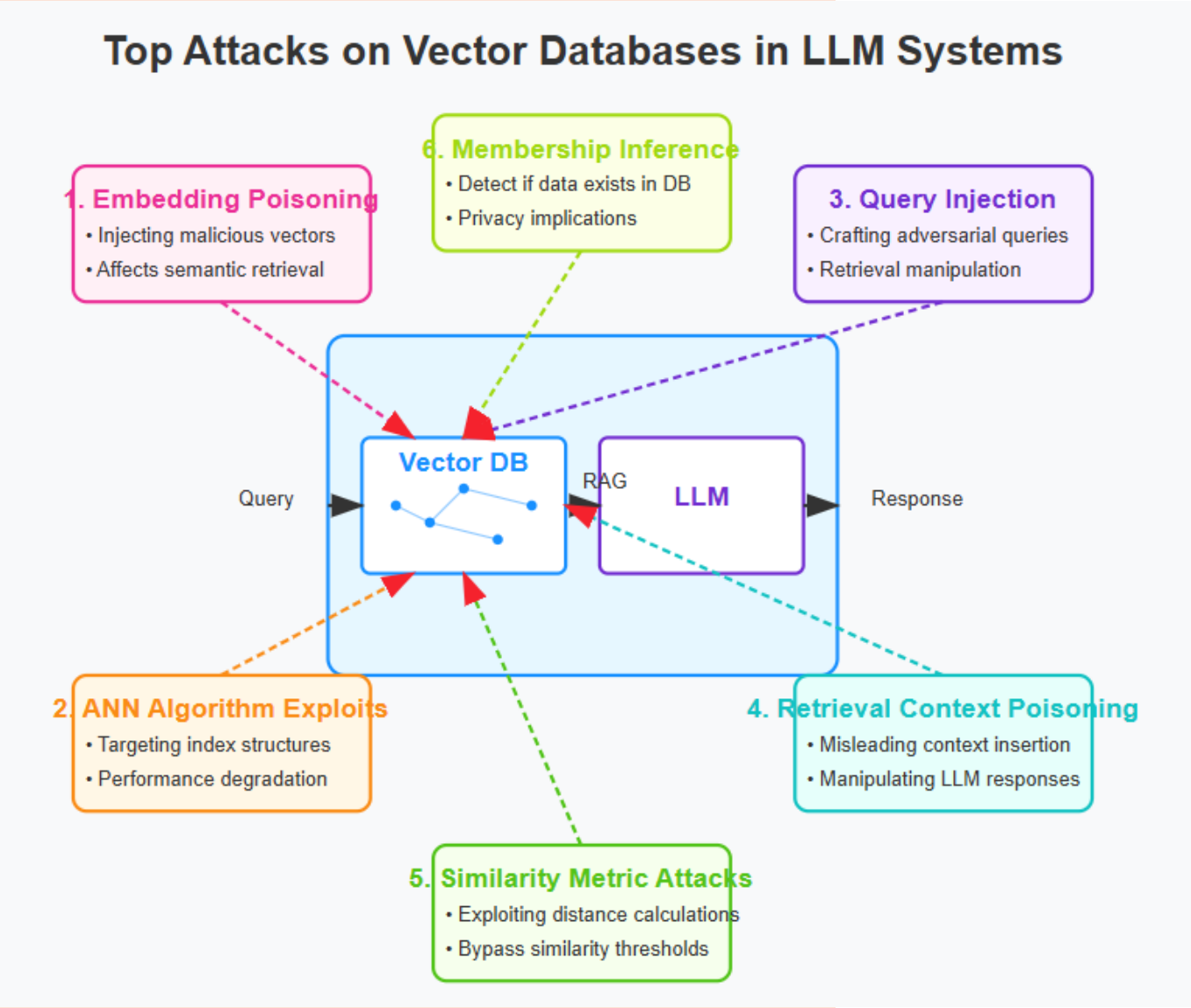
4. **Applications in GenAI:**
    - Retrieval-Augmented Generation (RAG)
    - Semantic search
    - Content recommendation
    - Deduplication
    - Knowledge management

Popular vector database options include Pinecone, Weaviate, Milvus, Qdrant, ChromaDB, and pgvector (PostgreSQL extension).



Harpreet Singh - Enterprise Security Architect

The Vector Database – Potential Top Attacks



1. **Embedding Poison Attacks** - Attackers inject maliciously crafted vectors into the database

2. **ANN Algorithm Exploitation -** Targets weaknesses in Approximate Nearest Neighbour algorithms used by vector databases

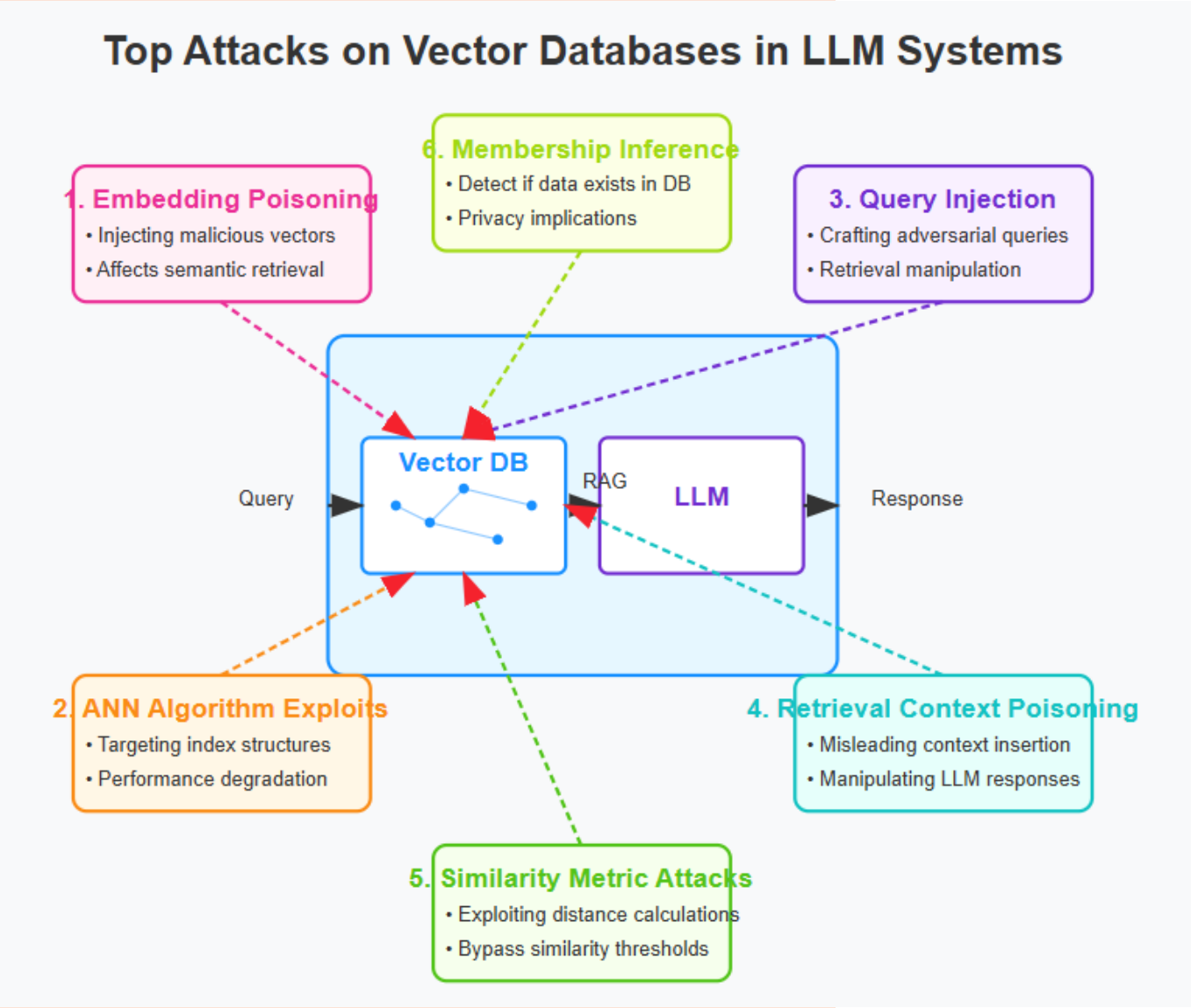
3. **Query Injection -** Manipulating input queries to retrieve specific harmful or misleading information

4. **Retrieval Context Poisoning -** Attacking the connection between vector retrieval and LLM context window

5. **Similarity Metric Attacks -** Exploiting how vector databases calculate similarity scores

6. **Membership Inference Attacks -** Determining whether specific data exists in the vector database

The Vector Database – Potential Top Attacks



Defense Strategies

1. **Input Validation**: Rigorously validate and sanitize vectors before insertion
2. **Anomaly Detection**: Monitor for unusual patterns in vector spaces
3. **Robust Similarity Thresholds**: Implement dynamic thresholds for retrieval
4. **Content Filtering**: Apply post-retrieval checks before passing to LLMs
5. **Access Controls**: Implement proper authentication/authorization for database operations
6. **Differential Privacy**: Add noise to vectors to prevent information leakage
7. **Regular Auditing**: Periodically verify the integrity of stored embeddings

Harpreet Singh - Enterprise Security Architect

The Vector Database – Potential Top Attacks

1. Embedding Poisoning Attacks

Description:

- Attackers inject maliciously crafted vectors into the database
- These poisoned embeddings can influence retrieval results when semantically similar queries are made

Impact:

- Can cause retrieval of inappropriate or harmful content
- May lead to LLM hallucinations or inaccurate responses
- Could potentially leak sensitive information

Example: Adding vectors with harmful content that are semantically similar to common benign queries.

2. ANN Algorithm Exploitation

Description:

- Targets weaknesses in Approximate Nearest Neighbor algorithms used by vector databases
- Exploits specific index structures like HNSW graphs or IVF partitions

Impact:

- May cause performance degradation (CPU/memory spikes)

The Vector Database – Potential Top Attacks

- Can result in reduced accuracy of search results
- Potential denial of service in extreme cases

Example: Crafting vectors that force excessive traversal of index structures, causing computational bottlenecks.

3. Query Injection

Description:

- Manipulating input queries to retrieve specific harmful or misleading information
- Creating adversarial embeddings designed to bypass safety filters

Impact:

- Retrieves unintended or unsafe content
- Can be used to jailbreak or circumvent safety measures
- May expose private information through carefully crafted queries

Example: Crafting a query vector that is designed to retrieve sensitive information while appearing innocuous.

4. Retrieval Context Poisoning

Description:

- Attacking the connection between vector retrieval and LLM context window
- Injecting misleading information that will be trusted by the LLM

The Vector Database – Potential Top Attacks

Impact:

- Can cause the LLM to generate incorrect or harmful responses
- Exploits the LLM's tendency to trust information in its context window
- Particularly effective in RAG systems

Example: Adding factually incorrect but plausible-sounding information that the LLM will incorporate into responses.

5. Similarity Metric Attacks

Description:

- Exploiting how vector databases calculate similarity scores
- Taking advantage of the chosen distance metric (cosine, euclidean, etc.)

Impact:

- Bypasses similarity thresholds used for filtering
- Can cause retrieval of unrelated but specifically crafted content
- May lead to inconsistent retrieval results

Example: Creating vectors engineered to have artificially high similarity scores with common queries despite semantic differences.

The Vector Database – Potential Top Attacks

6. Membership Inference Attacks

Description:

- Determining whether specific data exists in the vector database
- Using query response patterns to deduce database contents

Impact:

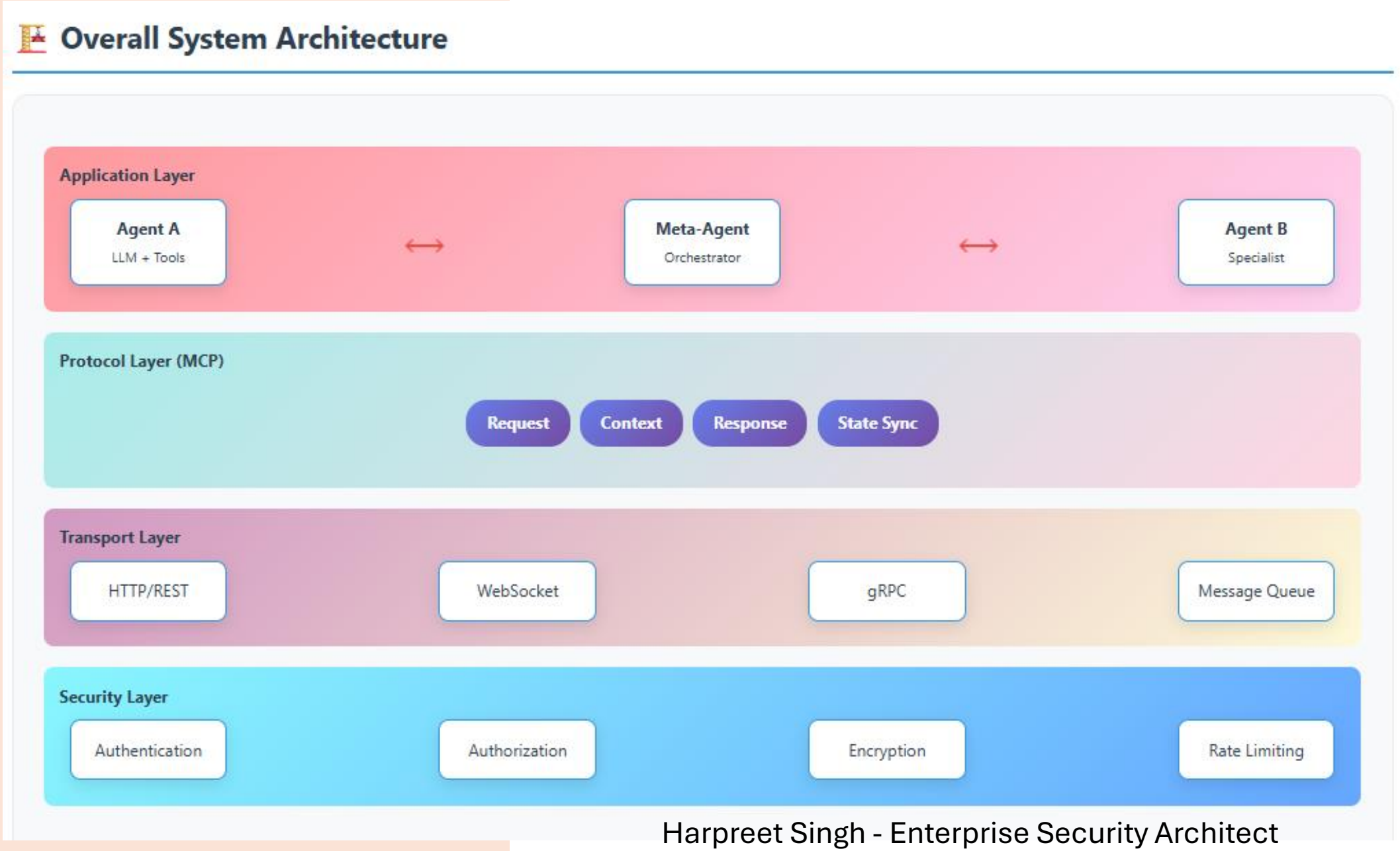
- Privacy violations by revealing what data is in the system
- Can expose sensitive or confidential information indirectly
- May lead to further targeted attacks

Example: Systematically probing the database with queries to determine if specific documents or information are present.

Agent 2 Agent vs MCP – The World of Protocols

Harpreet Singh - Enterprise Security Architect

Agent 2 Agent vs MCP – The World of Protocols



Agent 2 Agent vs MCP – The World of Protocols

🎯 Protocol Objectives

Context Continuity

Maintain conversation state and context across multiple agent interactions, ensuring coherent multi-turn dialogues.

Resource Coordination

Enable agents to share computational resources, data sources, and specialized capabilities efficiently.

Task Decomposition

Break complex queries into smaller tasks that can be distributed across specialized agents.

State Synchronization

Keep all participating agents synchronized with the current state of the conversation and shared context.

🔧 Model Context Protocol (MCP) Details

Core Components

```
{ "mcp_version": "1.0", "session_id": "uuid-session-123", "context": { "conversation_history": [...], "shared_memory": {...},
"agent_capabilities": {...} }, "message": { "type": "request|response|notification", "payload": {...}, "metadata": {...} } }
```

Context Management

- **Memory Persistence:** Long-term and short-term memory storage
- **Context Windows:** Managing token limits across agents
- **Relevance Filtering:** Selective context sharing
- **Compression:** Context summarization techniques

Message Routing

- **Agent Discovery:** Finding appropriate agents for tasks
- **Load Balancing:** Distributing requests efficiently
- **Fallback Mechanisms:** Handling agent failures
- **Priority Queuing:** Managing urgent vs. routine tasks

Agent 2 Agent vs MCP – The World of Protocols

Agent-to-Agent Communication & Model Context Protocol (MCP)

What is Agent-to-Agent Communication?

Agent-to-Agent (A2A) communication refers to the protocols and mechanisms that enable different AI agents to interact, share information, and coordinate tasks within a multi-agent system. This communication is essential for complex workflows that require multiple specialized agents working together.

What is Model Context Protocol (MCP)?

Model Context Protocol is a standardized communication framework that manages how context, state, and information flow between different AI agents and models. It ensures that all participating agents maintain a coherent understanding of the conversation, shared memory, and task progress.

Agent 2 Agent vs MCP – The World of Protocols

Why These Protocols Are Required

1. Complexity Management: Modern AI tasks often exceed the capabilities of a single agent, requiring specialized agents for different domains (e.g., data analysis, visualization, document creation).

2. Resource Optimization: Different agents can be optimized for specific tasks, reducing computational overhead and improving efficiency.

3. Scalability: Multi-agent systems can handle larger workloads by distributing tasks across multiple specialized components.

4. Fault Tolerance: If one agent fails, others can continue operating, with the system potentially routing around the failure.

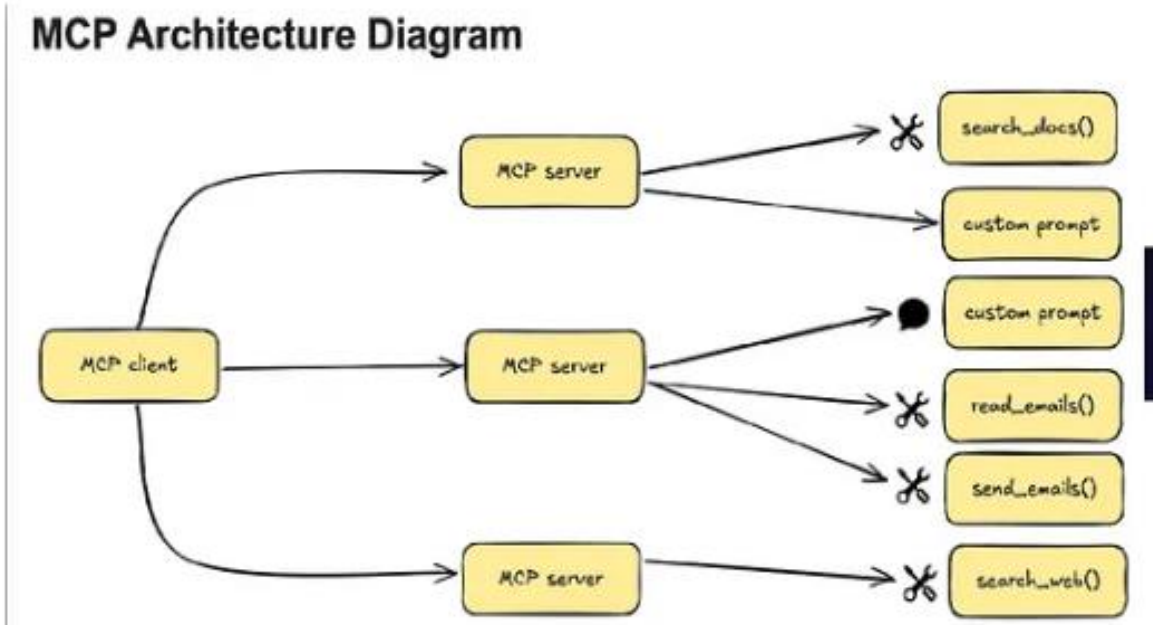
5. Context Continuity: Maintaining conversation state across multiple interactions and agents ensures coherent, contextually-aware responses.

Key Objectives Fulfilled

- **Context Preservation**: Ensuring that important information from previous interactions is available to all relevant agents
- **Task Coordination**: Synchronizing the work of multiple agents on complex, multi-step problems
- **Resource Sharing**: Enabling agents to access shared databases, APIs, and computational resources
- **Quality Assurance**: Implementing checks and balances through multiple agent validation
- **User Experience**: Providing seamless interactions despite complex backend orchestration

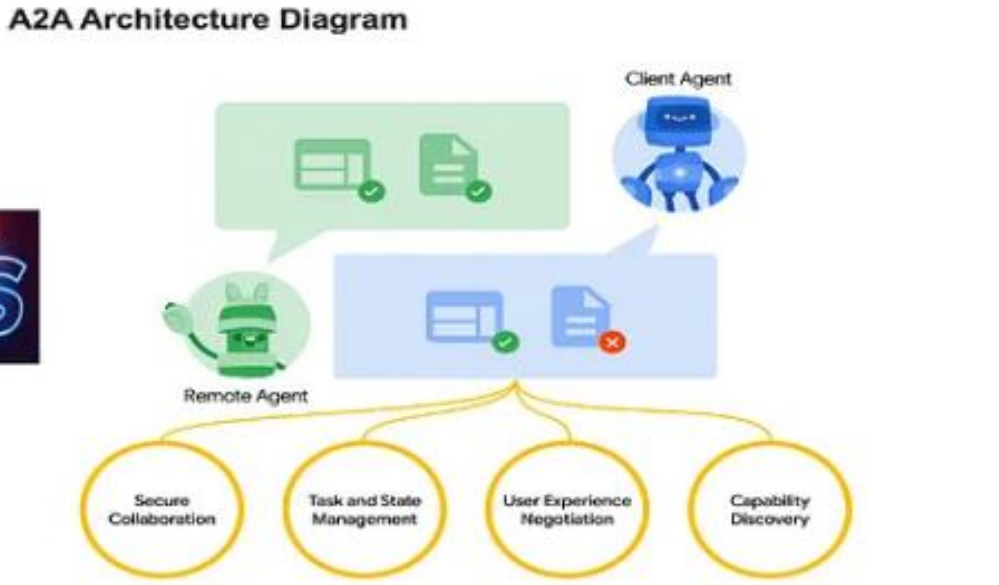
Harpreet Singh - Enterprise Security Architect

Agent 2 Agent vs MCP – The World of Protocols



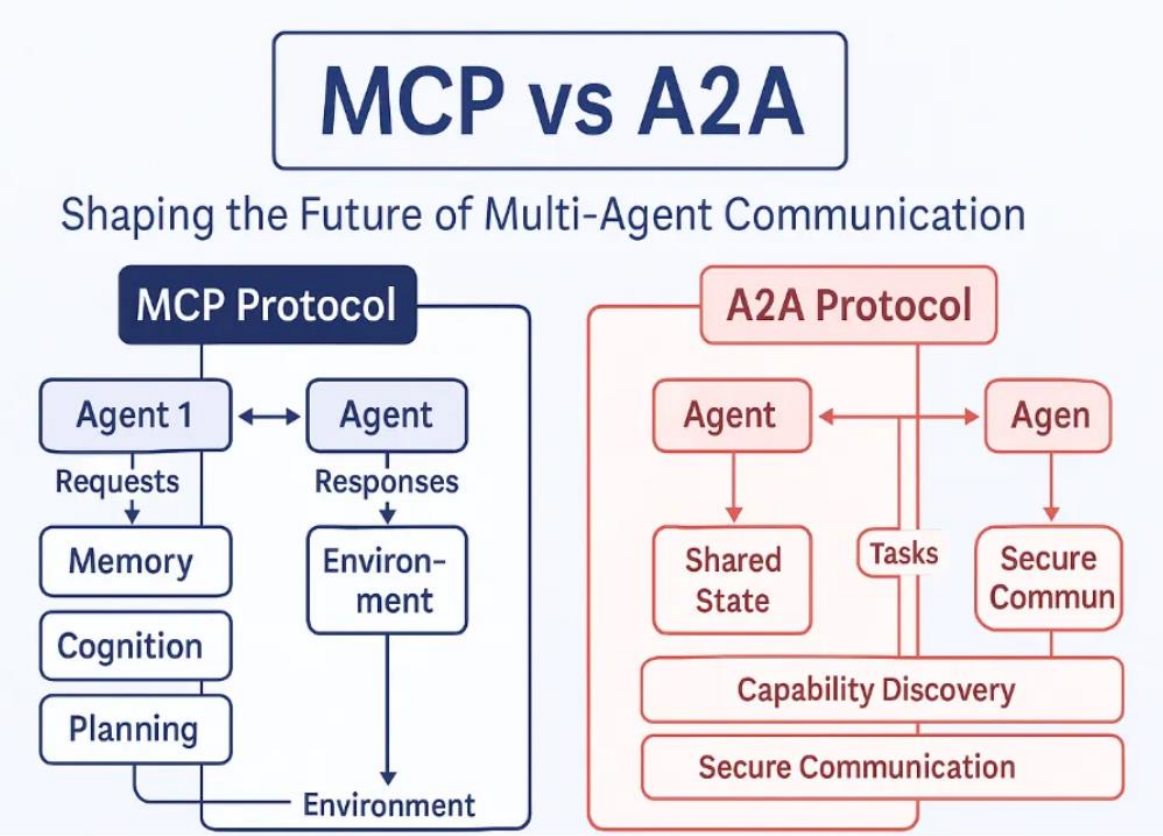
The diagram illustrates how an MCP client connects to multiple MCP servers, each providing different capabilities such as search functionality, custom prompts, email operations, and web search.

VS



The diagram shows how Client and Remote Agents interact, with four key capabilities highlighted: Secure Collaboration, Task and State Management, User Experience Negotiation, and Capability Discovery.

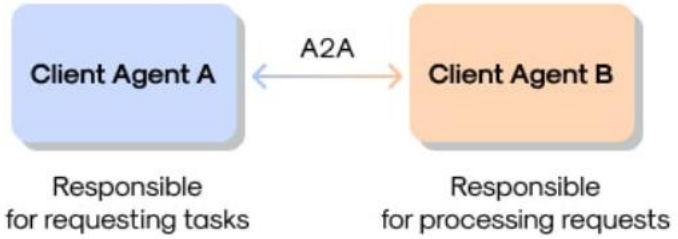
Agent 2 Agent vs MCP – The World of Protocols



A closer comparison of A2A and MCP reveals contrasting functionality.

	A2A	MCP
Functionality	Facilitates application-to-application data exchange	Oversees all communication via a central server
Security Verification	Trusts individual digital identities for application security	Relies on the central server for access authorization
Application	Primarily for real-time data exchange	Ensures superior oversight of all communication

How A2A Works



Agents can advertise their capabilities using Agent Cards in JSON format.

Communication between client and remote agents is oriented towards tasks.
Task objects is defined by A2A.

Agents send messages to communicate context, responses, artifacts, and user instructions.

Harpreet Singh - Enterprise Security Architect

Agent 2 Agent vs MCP – The World of Protocols

How A2A works with AI agents

A2A follows a client-server model where agents communicate over standard web protocols like HTTP using structured JSON messages. This approach improves compatibility with existing infrastructure while standardizing agent communication.

Let's explore how A2A accomplishes its goals by breaking down the components that make up the protocol, including the concept of "opaque" agents. Then, we'll take a closer look at a typical interaction flow.

Core components of A2A

Agent Card: Typically hosted at a well-known URL (like /.well-known/agent.json), this JSON file describes an agent's capabilities, skills, endpoint URL, and authentication requirements. Think of this as an agent's machine-readable "résumé" that helps other agents determine whether to engage with it. **A2A Server:** An agent that exposes HTTP endpoints that use the A2A protocol. This is the "remote agent" in A2A, which receives requests from the client agent and handles tasks. Servers advertise their capabilities via Agent Cards.

A2A Client: The app or AI system that consumes A2A services. The client constructs tasks and distributes them to the appropriate servers based on their capabilities and skills. This is the "client agent" in A2A, which orchestrates workflows with specialized servers.

Agent 2 Agent vs MCP – The World of Protocols

Task: The central unit of work in A2A. Each task has a unique ID and progresses through defined states (like submitted, working, completed, etc.). Tasks serve as containers for the work being requested and executed.

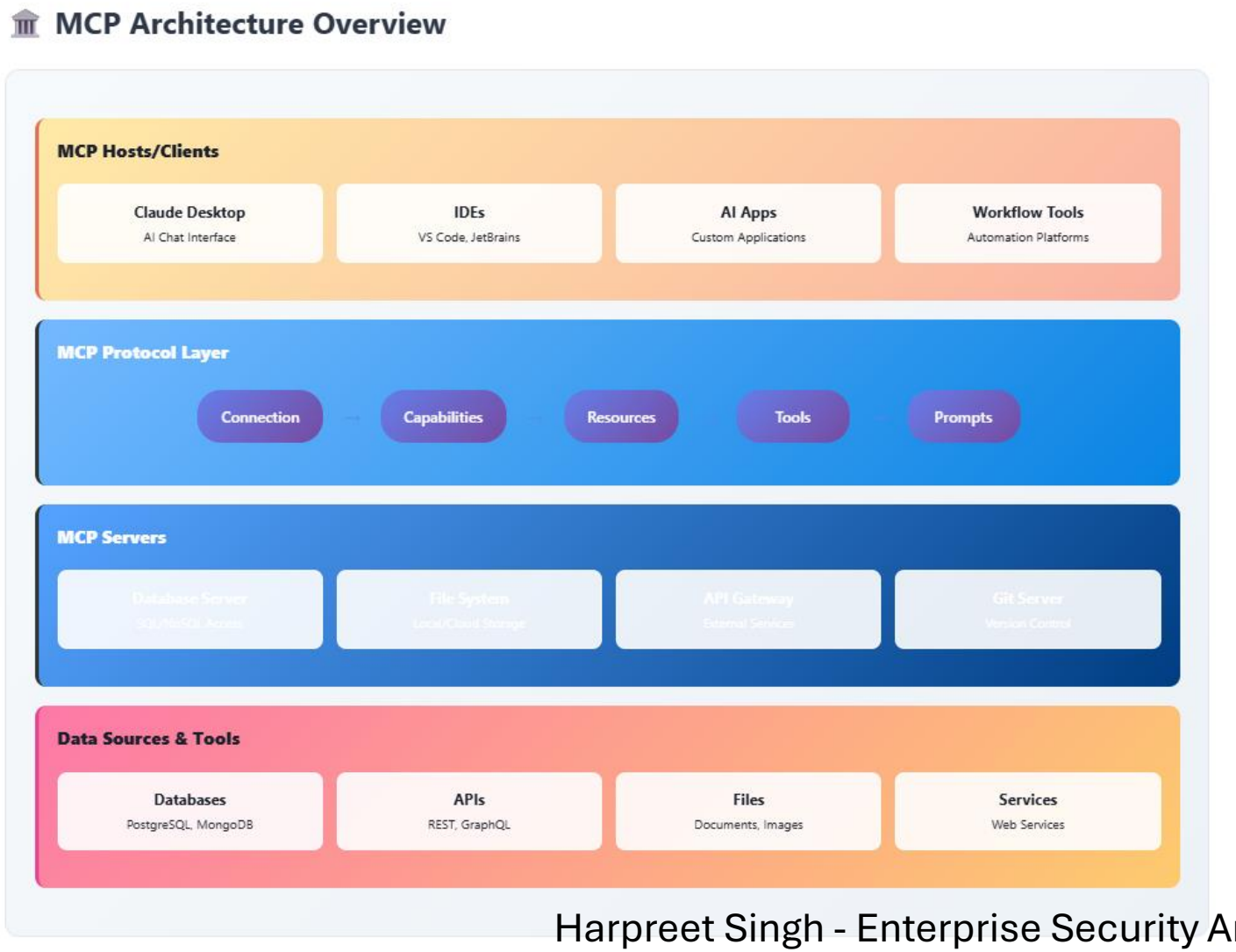
Message: Communication turns between the client and the agent. Messages are exchanged within the context of a task and contain Parts that deliver content.

Part: The fundamental content unit with a Message or Artifact. Parts can be:

- TextPart: For plain text or formatted content
- FilePart: For binary data (with inline bytes or a URI reference)
- DataPart: For structured JSON data (like forms)

Artifact: The output generated by an agent during a task. Artifacts also contain Parts and represent the final deliverable from the server back to the client.

Agent 2 Agent vs MCP – The World of Protocols



Agent 2 Agent vs MCP – The World of Protocols

⚙️ How MCP Works

Connection Flow

1. **Initialize** Client connects to MCP server via transport (stdio, SSE, WebSocket)
2. **Capabilities** Server advertises available resources, tools, and prompts
3. **Discovery** Client queries for specific resources and tools
4. **Execution** Client invokes tools and retrieves resources as needed
5. **Response** Server returns data/results back to client

Core Protocol Components

```
{ "jsonrpc": "2.0", "method": "initialize", "params": { "protocolVersion": "2024-11-05", "capabilities": { "resources": { "subscribe": true, "listChanged": true }, "tools": { "listChanged": true }, "prompts": { "listChanged": true } }, "clientInfo": { "name": "Claude Desktop", "version": "1.0.0" } } }
```

🔌 Transport Protocols

- **stdio:** Standard input/output
- **SSE:** Server-Sent Events
- **WebSocket:** Real-time bidirectional

📦 Message Types

- **Request:** Client → Server
- **Response:** Server → Client
- **Notification:** One-way messages

🛠️ Core Primitives

- **Resources:** Data access
- **Tools:** Function calls
- **Prompts:** Template management

MCP – Used Cases

🎯 Real-World Use Cases

💼 Business Intelligence & Analytics

Scenario: AI assistant analyzes sales data from multiple databases
MCP Role: Connects to PostgreSQL, Salesforce API, and Excel files simultaneously
Benefit: Single query retrieves and correlates data from all sources

💻 Software Development

Scenario: AI coding assistant helps debug and refactor code
MCP Role: Accesses Git repositories, documentation, and testing frameworks
Benefit: Context-aware code suggestions with access to entire project history

📊 Data Science Workflows

Scenario: AI helps with data exploration and model building
MCP Role: Connects to data lakes, ML platforms, and visualization tools
Benefit: Seamless data pipeline from raw data to insights

🏥 Healthcare Information Systems

Scenario: AI assistant helps clinicians with patient care
MCP Role: Securely accesses EHR systems, medical databases, and research papers
Benefit: Comprehensive patient information with latest medical research

🏦 Financial Services

Scenario: AI analyzes market data for investment decisions
MCP Role: Connects to trading platforms, news feeds, and risk management systems
Benefit: Real-time market analysis with comprehensive risk assessment

🛒 E-commerce & Customer Service

Scenario: AI chatbot provides customer support
MCP Role: Accesses inventory systems, order databases, and knowledge bases
Benefit: Personalized customer service with real-time information

MCP – Used Cases

🔧 Implementation Example

Basic MCP Server Setup

```
import { Server } from "@modelcontextprotocol/sdk/server/index.js"; import { StdioServerTransport } from "@modelcontextprotocol/sdk/server/stdio.js"; const server = new Server( { name: "database-server", version: "1.0.0", }, { capabilities: { resources: {}, tools: {}, }, } ); // Define a resource server.setRequestHandler(ListResourcesRequestSchema, async () => { return { resources: [ { uri: "database://users", name: "User Database", description: "Access to user information", mimeType: "application/json", }, ], }; }); // Define a tool server.setRequestHandler(CallToolRequestSchema, async (request) => { if (request.params.name === "query_database") { const { query } = request.params.arguments; const result = await executeQuery(query); return { content: [ { type: "text", text: JSON.stringify(result, null, 2), }, ], }; } }); // Start server const transport = new StdioServerTransport(); await server.connect(transport);
```

Client Configuration (Claude Desktop)

```
{ "mcpServers": { "database": { "command": "node", "args": ["database-server.js"], "env": { "DATABASE_URL": "postgresql://localhost/mydb" } }, "filesystem": { "command": "npx", "args": ["-y", "@modelcontextprotocol/server-filesystem", "/path/to/files"] } } }
```

📋 Summary

🎯 Purpose

Standardized protocol for AI applications to connect with external data sources and tools, eliminating custom integrations.

🏗️ Architecture

Client-server model with JSON-RPC protocol, supporting multiple transport methods and standardized message formats.

⚡ Key Benefits

Simplified integrations, enhanced security, better scalability, and improved developer experience across AI applications.

🌐 Ecosystem

Growing ecosystem of MCP servers, client implementations, and tools supporting various data sources and services.

🚀 The Future of AI Integration

MCP represents a paradigm shift in how AI applications interact with the world. By providing a standardized, secure, and scalable protocol, it enables developers to build more powerful AI applications while reducing complexity and maintenance overhead.

Agent 2 Agent vs MCP – The World of Protocols

⚠️ Security Attack Vectors

🎭 Agent Impersonation Attacks

Description: Malicious actors create fake agents that mimic legitimate ones to intercept or manipulate communications.

Impact: Data theft, misinformation injection, unauthorized access to sensitive operations.

🧠 Context Poisoning

Description: Injection of malicious or biased information into shared context memory to influence agent decision-making.

Impact: Biased outputs, incorrect decisions, propagation of misinformation across agent network.

💥 Protocol Manipulation

Description: Exploiting weaknesses in MCP message format or routing logic to cause system failures or gain unauthorized access.

Impact: System crashes, unauthorized command execution, data corruption.

🔄 Infinite Loop Attacks

Description: Crafting requests that cause agents to enter infinite communication loops, consuming resources.

Impact: Denial of service, resource exhaustion, system instability.

🕵️ Eavesdropping & MITM

Description: Intercepting communications between agents to steal sensitive information or modify messages in transit.

Impact: Data breaches, message tampering, unauthorized access to confidential information.

⚡ Resource Exhaustion

Description: Overwhelming agents with excessive requests or large context payloads to cause performance degradation.

Impact: Service disruption, increased costs, system failures.

Agent 2 Agent vs MCP – The World of Protocols

🛡️ Security Countermeasures

Authentication & Authorization

- Multi-factor agent authentication
- Role-based access control (RBAC)
- Digital certificates for agent identity
- Token-based authorization

Communication Security

- End-to-end encryption (TLS 1.3+)
- Message integrity verification
- Secure key exchange protocols
- Certificate pinning

Context Protection

- Input validation and sanitization
- Context integrity checksums
- Anomaly detection in context data
- Context access logging and auditing

System Resilience

- Rate limiting and throttling
- Circuit breaker patterns
- Resource quotas and limits
- Health monitoring and alerting

🔄 Protocol Flow Example

```
1. User Query → Meta-Agent ├─ Query: "Analyze sales data and create presentation" └─ Context: User preferences, available data sources 2. Meta-Agent → Task Decomposition ├─ Task 1: Data Analysis (routes to Analytics Agent) ├─ Task 2: Visualization (routes to Chart Agent) └─ Task 3: Presentation Creation (routes to Document Agent) 3. Agent Communication (MCP) ├─ Analytics Agent ↔ Database Agent (data retrieval) ├─ Chart Agent ← Analytics Agent (processed data) └─ Document Agent ← Chart Agent (visualizations) 4. Context Synchronization ├─ Shared memory updates with intermediate results ├─ Progress tracking across all agents └─ Error handling and rollback mechanisms 5. Final Assembly ├─ Meta-Agent consolidates all outputs ├─ Quality assurance checks └─ Response delivery to user
```

Testing AI System Security

Testing AI System Security

Testing an AI system's security relies on three strategies:

- **Conventional security testing** (i.e. *pentesting*). See secure software development.

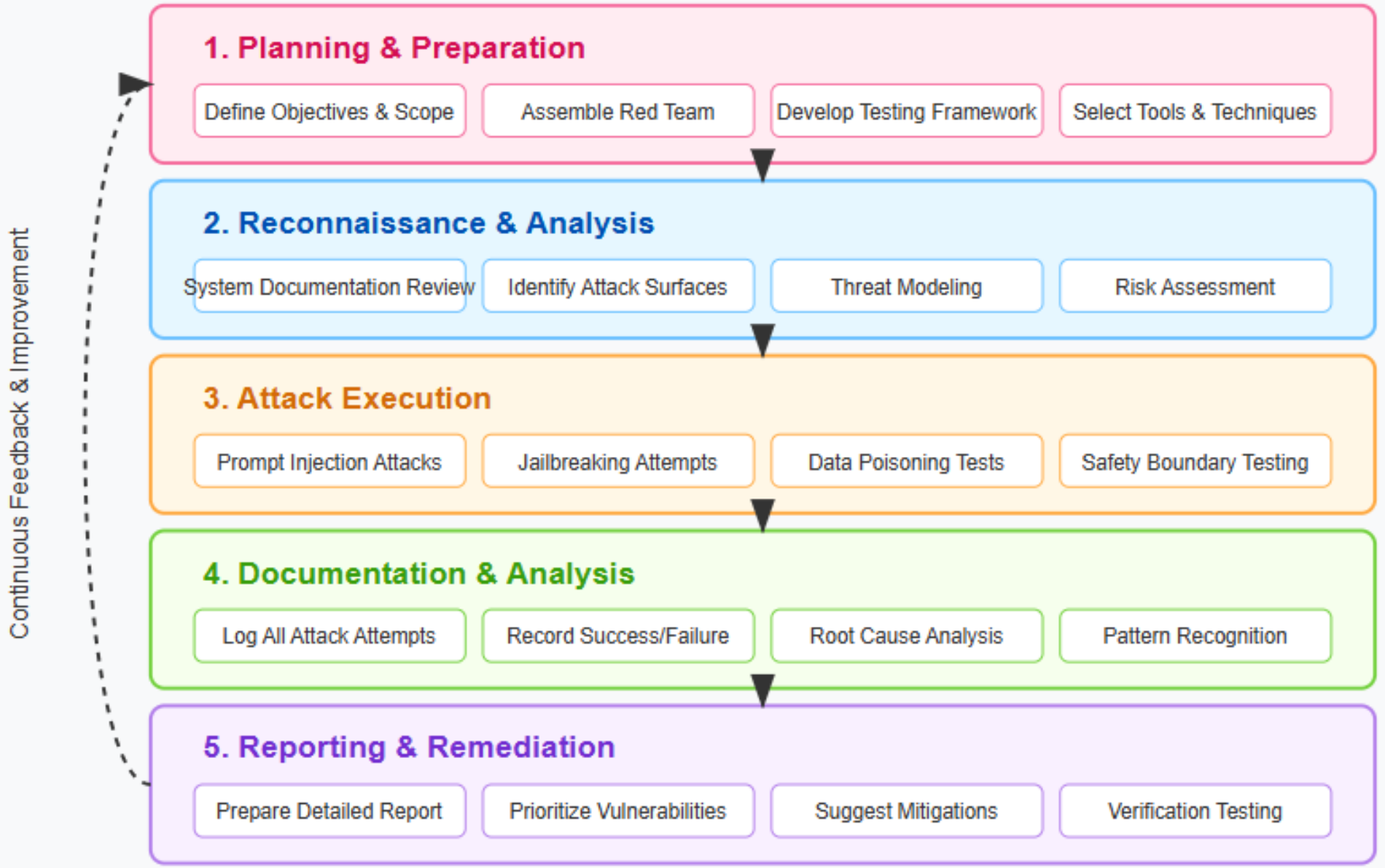
- **Model performance validation** (see continuous validation): testing if the model behaves according to its specified acceptance criteria using a validation set with inputs and outputs that represent the intended behaviour of the model. For security, this is to detect if the model behaviour has been altered permanently through data poisoning or model poisoning. For non-security, it is for testing functional correctness, model drift etc.

- **AI security testing** (this section), the part of *AI red teaming* that tests if the AI model can withstand certain attacks, by simulating these attacks.
- AI security tests simulate adversarial behaviors to uncover vulnerabilities, weaknesses, and risks in AI systems. While the focus areas of traditional AI testing are functionality and performance, the focus areas of AI Red Teaming go beyond standard validation and include intentional stress testing, attacks, and attempts to bypass safeguards. While the focus of red teaming can extend beyond Security, in this document, we focus primarily on "AI Red Teaming for AI Security".
- In this section, we differentiate AI Red Teaming for Predictive and Generative AI due to their distinct nature, risks, and applications. While some threats, such as development-time supply chain threats, could be common to both types of AI, the way they manifest in their applications can differ significantly.

AI Red Teaming (Systematic Approach)

- **Define Objectives and Scope**: Identification of objectives, alignment with organizational, compliance, and risk management requirements.
- **Understand the AI System:** Details about the model, use cases, and deployment scenarios.
- **Identify Potential Threats:** Threat modeling, identification of attack surface, exploration, and threat actors.
- **Develop Attack Scenarios:** Design of attack scenarios and edge cases.
- **Test Execution:** Conduct manual or automated tests for the attack scenarios.
- **Risk Assessment:** Documentation of the identified vulnerabilities and risks.
- **Prioritization and Risk Mitigation:** Develop an action plan for remediation, implement mitigation measures, and calculate residual risk.
- **Validation of Fixes:** Retest the system post-remediation.

AI Red Teaming: Systematic Approach



1. Planning & Preparation

Define Objectives & Scope

- Clearly outline the testing objectives and success criteria
- Determine which AI system components will be tested
- Set boundaries for permissible testing activities
- Establish timelines and resource allocation

Assemble Red Team

- Recruit specialists with diverse expertise (prompt engineering, ML security, cybersecurity)
- Include domain experts relevant to the AI application
- Consider both technical and non-technical perspectives
- Establish team roles and responsibilities

Develop Testing Framework

- Create methodologies for systematic testing
- Develop evaluation metrics and scoring systems
- Define success criteria and benchmarks
- Establish documentation standards

Select Tools & Techniques

- Choose appropriate testing tools and platforms
- Prepare testing infrastructure
- Develop custom attack scripts or tools if needed
- Create sandbox environments for safe testing

2. Reconnaissance & Analysis

System Documentation Review

- Examine API documentation and model specifications
- Review training data sources when available
- Understand system architecture and deployment
- Analyse previous security assessments

Identify Attack Surfaces

- Map all input channels and user interaction points
- Identify potential vulnerability points in the AI pipeline
- Assess integration points with other systems
- Evaluate data flow and processing pathways

Threat Modelling

- Create attack trees and scenarios
- Identify relevant threat actors and their motivations
- Apply frameworks like STRIDE or MITRE ATT&CK for AI
- Determine attack vectors specific to the AI system

Risk Assessment

- Prioritize vulnerabilities based on impact and likelihood
- Analyse potential consequences of successful attacks
- Consider both technical and business impacts
- Identify high-risk areas for focused testing

3. Attack Execution

Prompt Injection Attacks

- Test direct and indirect prompt injection techniques
- Attempt context manipulation
- Try multi-step prompt injection chains
- Test resilience against different injection patterns

Jailbreaking Attempts

- Test known jailbreak techniques and templates
- Develop custom jailbreaks for the specific model
- Try multi-modal jailbreaking when applicable
- Test bypassing of safety mechanisms

Data Poisoning Tests

- Simulate training data poisoning scenarios
- Test automatic learning vulnerabilities
- Attempt to manipulate model behaviour through inputs
- Evaluate model robustness against adversarial example

Safety Boundary Testing

- Test model responses to harmful content requests
- Probe for sensitive information disclosure
- Test for bias exploitation and amplification
- Evaluate alignment drift under various conditions

4. Documentation & Analysis

Log All Attack Attempts

- Document all test cases with detailed methodology
- Record exact inputs and system responses
- Maintain secure records of all testing activities
- Preserve evidence of vulnerabilities

Record Success/Failure

- Classify attack outcomes (success, partial success, failure)
- Document unexpected behaviours even if not successful attacks
- Create reproducible test cases for verified vulnerabilities
- Track attack success rates across different categories

Root Cause Analysis

- Determine underlying causes of vulnerabilities
- Identify patterns in successful attacks
- Trace failure modes to specific model components
- Analyse why certain defences failed

Pattern Recognition

- Identify common vulnerability patterns
- Look for correlations between different attack types
- Analyze trends in system responses
- Detect systemic weaknesses

5. Reporting & Remediation

Prepare Detailed Report

- Create comprehensive documentation of findings
- Include executive summary for leadership
- Provide technical details for engineering teams
- Develop visual aids to communicate complex issues

Prioritize Vulnerabilities

- Rank issues by severity and exploitability
- Classify vulnerabilities by type and impact
- Recommend addressing order based on risk
- Consider dependencies between vulnerabilities

Suggest Mitigations

- Propose specific remediation strategies
- Include both short-term fixes and long-term solutions
- Recommend defensive measures and monitoring
- Provide implementation guidance where appropriate

Verification Testing

- Retest after mitigation implementation
- Verify effectiveness of security improvements
- Conduct regression testing to ensure no new vulnerabilities
- Document improvement metrics

Key Principles for Effective AI Red Teaming

1. **Adversarial Mindset**: Think like an attacker while maintaining ethical boundaries
2. **Systematic Approach**: Use structured methodologies rather than ad-hoc testing
3. **Documentation Focus**: Thoroughly document all activities for reproducibility
4. **Continuous Improvement**: Treat red teaming as an ongoing process, not a one-time event
5. **Cross-functional Collaboration**: Involve both AI experts and security specialists
6. **Ethical Considerations**: Maintain responsible disclosure practices
7. **Realistic Scenarios**: Test under conditions that mirror real-world usage

This systematic approach ensures comprehensive coverage of potential vulnerabilities while providing actionable intelligence for improving AI system security and safety.

Key Threats to Predictive AI:

•Evasion Attacks: These attacks occur when an attacker crafts inputs that mislead the model, causing it to perform its task incorrectly.

•Model Theft: In this attack, the model's parameters or functionality are stolen. This enables the attacker to create a replica model, which can then be used as an oracle for crafting adversarial attacks and other compounded threats.

•Model Poisoning: This involves the manipulation of data, the data pipeline, or the model training supply chain during the training phase (development phase). The attacker's goal is to alter the model's behavior which could result in undesired model operation.

Generative AI: Generative AI systems produce outputs such as text, images, or audio. Examples include large language models (LLMs) like ChatGPT and large vision models (LVMs) like DALL-E and MidJourney.

Key Threats to Generative AI:

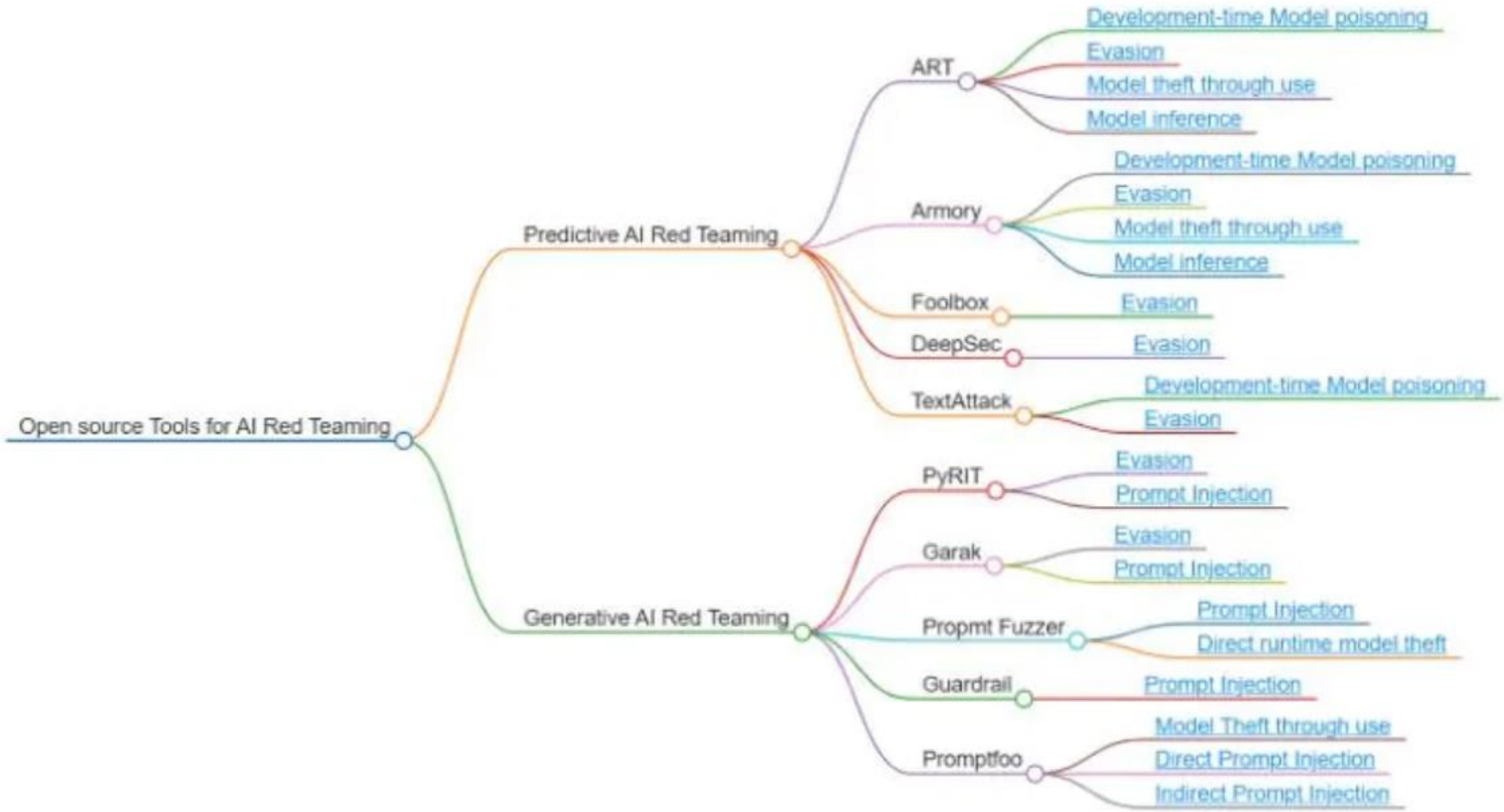
•Prompt Injection: In this type of attack, the attacker provides the model with manipulative instructions aimed at achieving malicious outcomes or objectives.

•Direct Runtime Model Theft: Attackers target parts of the model or critical components like the system prompt. By doing so, they gain the ability to craft sophisticated inputs that bypass guardrails.

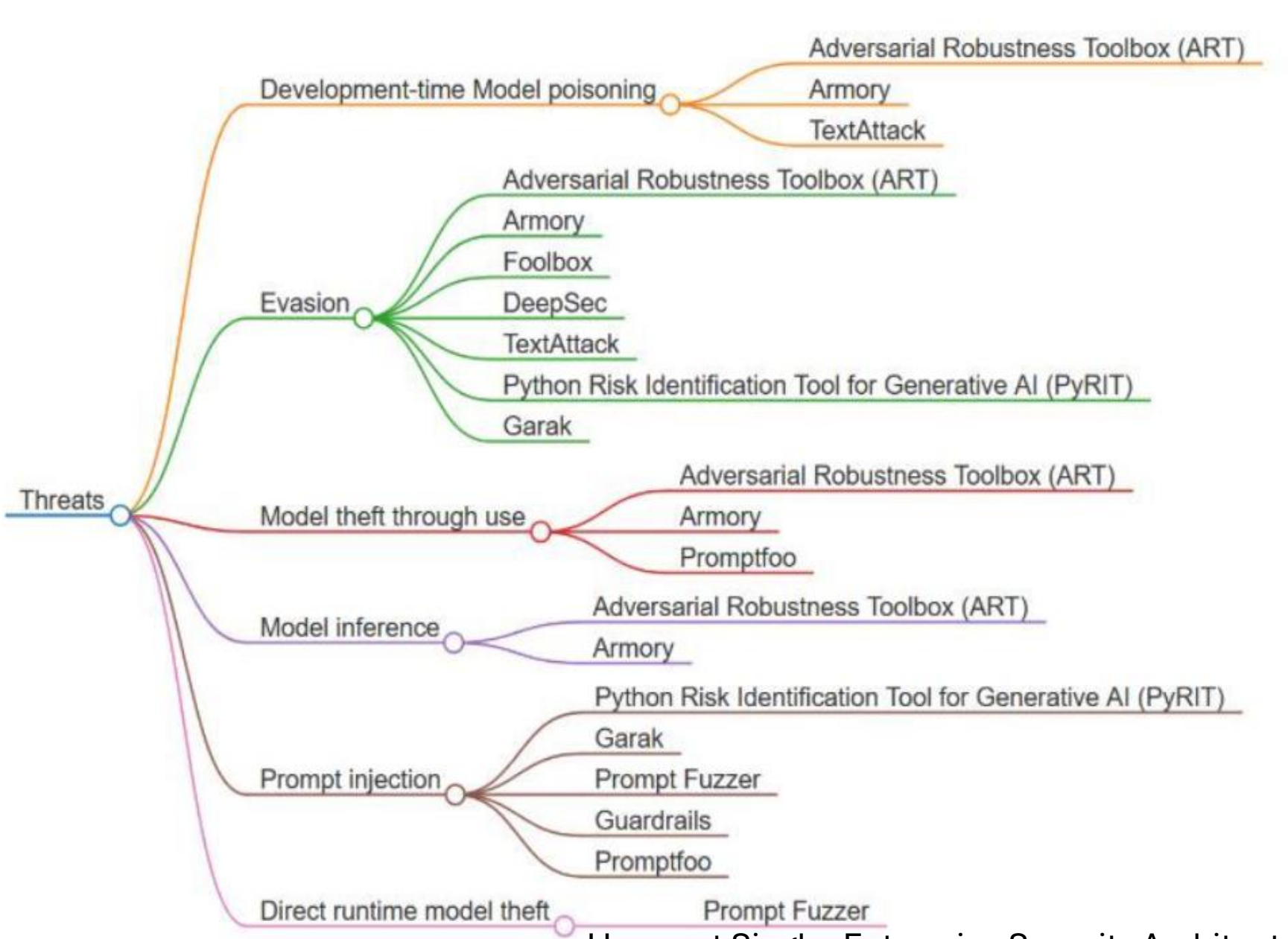
•Insecure Output Handling: Generative AI systems can be vulnerable to traditional injection attacks, leading to risks if the outputs are improperly handled or processed.

While we have mentioned the key threats for each of the AI Paradigm, we strongly encourage the reader to refer to all threats at the AI Exchange, based on the outcome of the Objective and scope definition phase in AI Red Teaming.

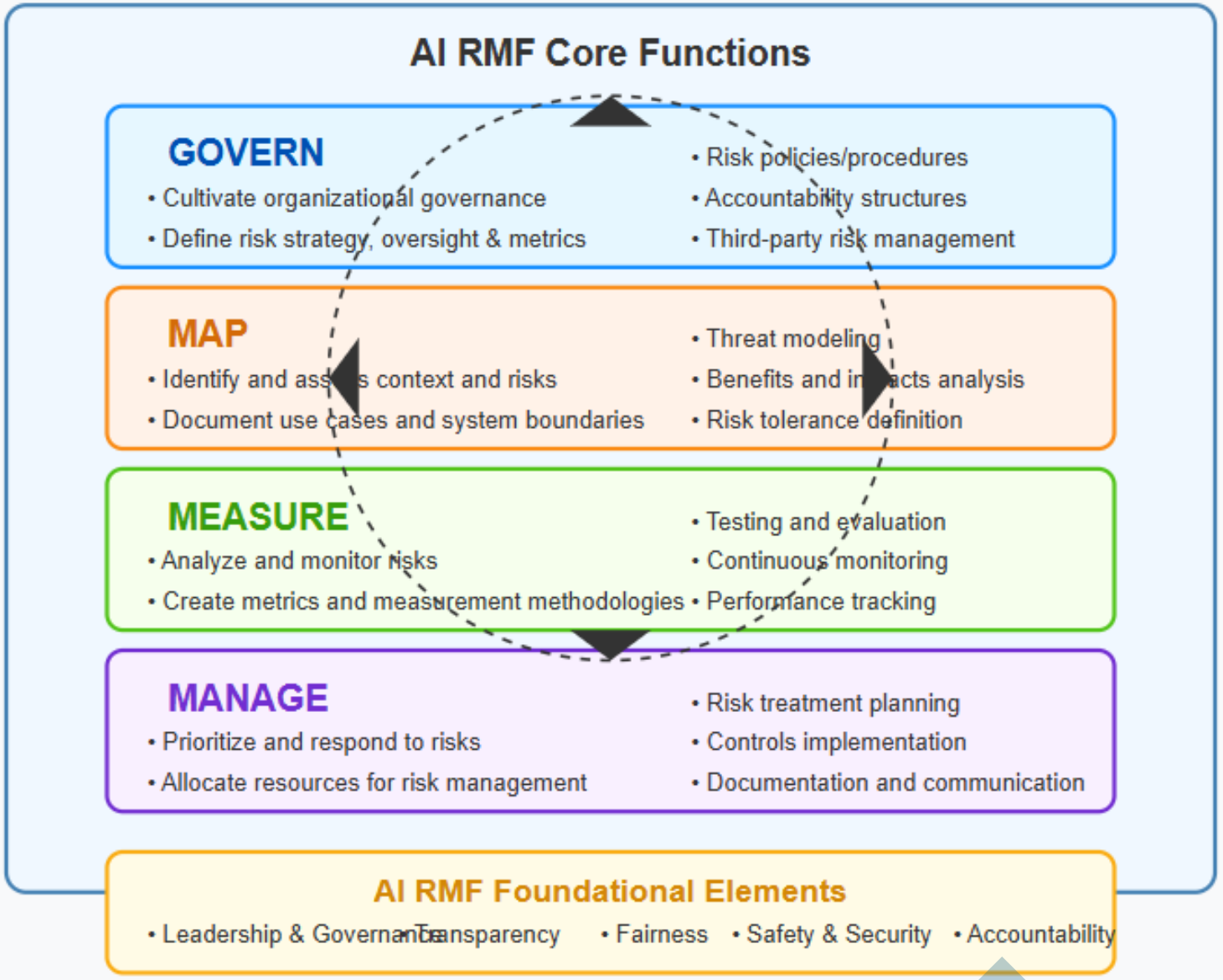
Red Teaming Tools for AI and GenAI



The diagram below categorizes threats in AI systems and maps them to relevant open-source tools designed to address these threats.



NIST AI Risk Management Framework



Core Functions of the NIST AI RMF

The framework is organized around four key functions that work together in a continuous cycle:

1. GOVERN

Definition: Establish and implement a governance structure to manage AI-related risks.

Key Activities:

- Develop and implement an AI risk management strategy and program
- Define roles, responsibilities, and accountabilities
- Establish policies and procedures for AI risk management
- Ensure organizational alignment between business objectives and risk tolerance
- Allocate resources for AI risk management activities
- Create oversight mechanisms for internal and third-party AI systems

2. MAP

Definition: Identify, analyse, and document AI system context, capabilities, and associated risks.

Key Activities:

- Identify AI system context, purpose, and intended use cases
- Document system capabilities, limitations, and dependencies
- Map potential impacts on individuals, groups, organizations, and society
- Conduct comprehensive risk assessments and impact analyses
- Identify relevant laws, regulations, and standards
- Engage stakeholders in the mapping process

3. MEASURE

Definition: Establish metrics and methods to evaluate AI risks and risk management effectiveness.

Key Activities:

- Define appropriate metrics for measuring AI risks and impacts
- Develop methodologies for testing and evaluating AI systems
- Conduct regular assessments against pre-defined thresholds
- Implement monitoring mechanisms for deployed AI systems
- Track performance regarding trustworthiness characteristics
- Validate effectiveness of risk controls and mitigation strategies

4. MANAGE

Definition: Allocate resources to manage AI risks according to the organization's risk profile and priorities.

Key Activities:

- Determine appropriate risk responses (accept, avoid, mitigate, transfer, share)
- Design and implement controls to address identified risks
- Create incident response and recovery plans
- Document risk management decisions and their rationale
- Communicate with stakeholders about risk management activities
- Continuously improve risk management practices

Foundational Elements of the AI RMF

These elements underpin all four core functions and represent characteristics of trustworthy AI:

1. **Transparency:** Providing clear information about AI operation and decision-making
2. **Fairness:** Minimizing harmful bias and discriminatory outcomes
3. **Accountability:** Ensuring responsibility for AI actions and decisions
4. **Safety & Security:** Protecting against unintended harm and unauthorized access
5. **Explainability:** Enabling understanding of how and why AI systems reach outcomes
6. **Privacy:** Protecting personal information and respecting data rights
7. **Validity & Reliability:** Ensuring AI systems operate as intended with consistent results
8. **Human-Centered Values:** Aligning AI systems with human values and expectations

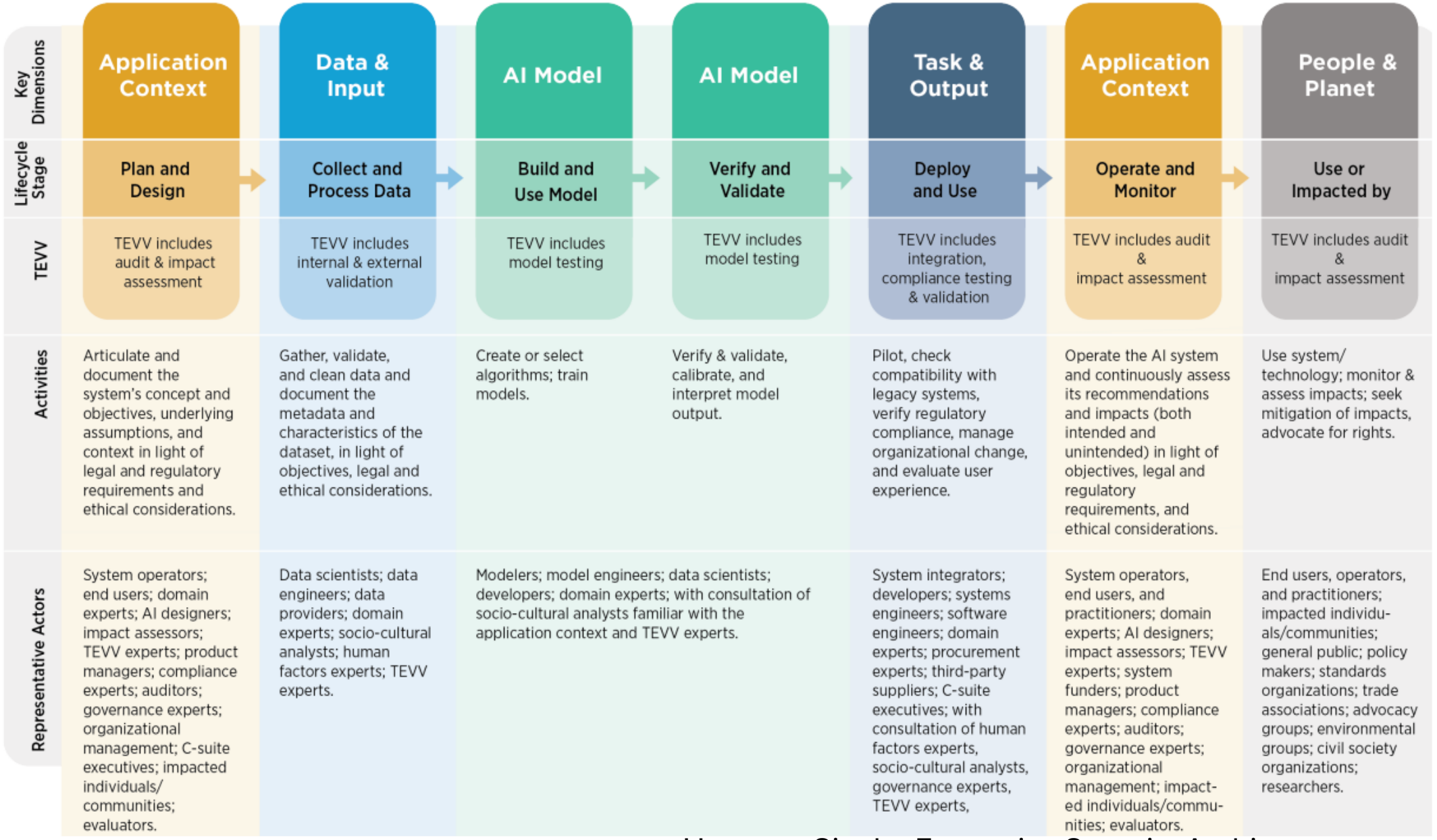
Implementation of the NIST AI RMF

Organizations typically implement the framework through these steps:

1. **Organizational Integration:** Incorporate AI risk management into existing governance structures
2. **Risk Prioritization:** Focus on highest-impact AI systems and most significant risks
3. **Proportionality:** Scale risk management efforts based on context and potential impact
4. **Continuous Improvement:** Regularly review and update risk management practices
5. **Stakeholder Engagement:** Involve relevant parties throughout the risk management process
6. **Documentation:** Maintain comprehensive records of risk management activities

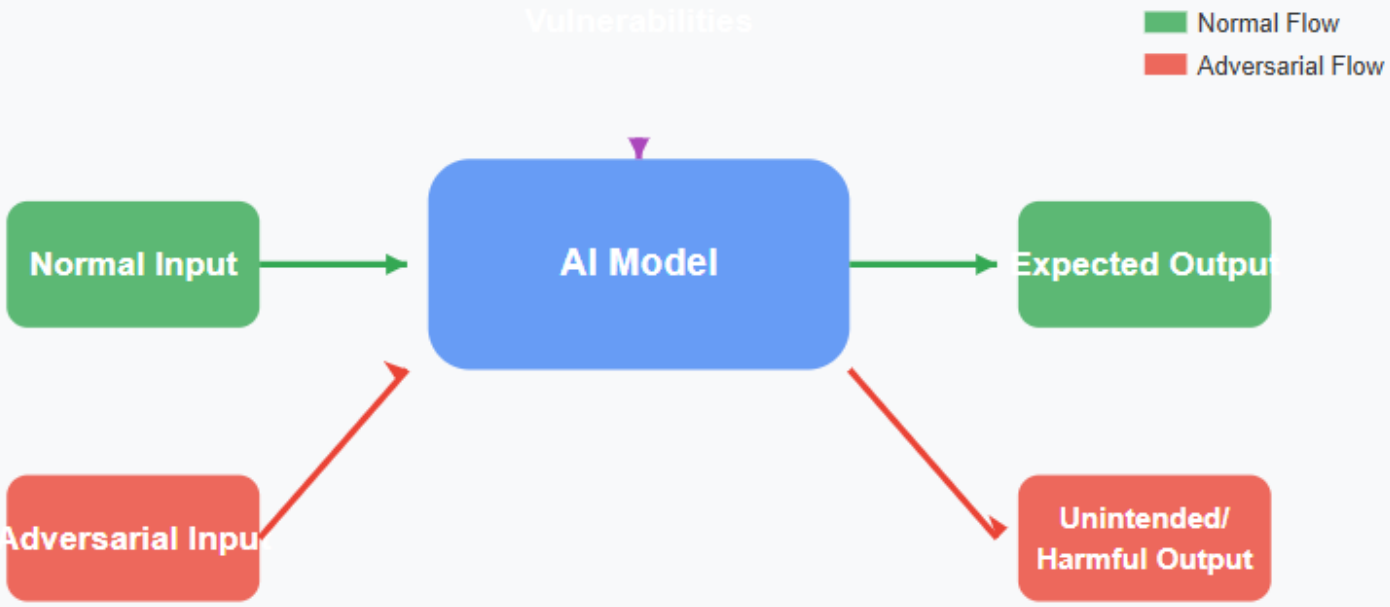
The NIST AI RMF is designed to be flexible and adaptable to various organizations, sectors, and AI applications, providing a structured approach to managing AI risks while fostering innovation and maintaining societal trust in AI technologies.


Plan and Design
Collect and process data
Operate and Monitor
Application Context
Data and Input
People and Planet
Task and Output
AI Model
Deploy and Use
Verify and validate
Build and use model



Key Dimensions	Application Context	Data & Input	AI Model	AI Model	Task & Output	Application Context	People & Planet
Lifecycle Stage	Plan and Design	Collect and Process Data	Build and Use Model	Verify and Validate	Deploy and Use	Operate and Monitor	Use or Impacted by
TEVV	TEVV includes audit & impact assessment	TEVV includes internal & external validation	TEVV includes model testing	TEVV includes model testing	TEVV includes integration, compliance testing & validation	TEVV includes audit & impact assessment	TEVV includes audit & impact assessment
Activities	Articulate and document the system's concept and objectives, underlying assumptions, and context in light of legal and regulatory requirements and ethical considerations.	Gather, validate, and clean data and document the metadata and characteristics of the dataset, in light of objectives, legal and ethical considerations.	Create or select algorithms; train models.	Verify & validate, calibrate, and interpret model output.	Pilot, check compatibility with legacy systems, verify regulatory compliance, manage organizational change, and evaluate user experience.	Operate the AI system and continuously assess its recommendations and impacts (both intended and unintended) in light of objectives, legal and regulatory requirements, and ethical considerations.	Use system/ technology; monitor & assess impacts; seek mitigation of impacts, advocate for rights.
Representative Actors	System operators; end users; domain experts; AI designers; impact assessors; TEVV experts; product managers; compliance experts; auditors; governance experts; organizational management; C-suite executives; impacted individuals/ communities; evaluators.	Data scientists; data engineers; data providers; domain experts; socio-cultural analysts; human factors experts; TEVV experts.	Modelers; model engineers; data scientists; developers; domain experts; with consultation of socio-cultural analysts familiar with the application context and TEVV experts.		System integrators; developers; systems engineers; software engineers; domain experts; procurement experts; third-party suppliers; C-suite executives; with consultation of human factors experts, socio-cultural analysts, governance experts, TEVV experts,	System operators, end users, and practitioners; domain experts; AI designers; impact assessors; TEVV experts; system funders; product managers; compliance experts; auditors; governance experts; organizational management; impact-ed individuals/commu-nities; evaluators.	End users, operators, and practitioners; impacted individu-als/communities; general public; policy makers; standards organizations; trade associations; advocacy groups; environmental groups; civil society organizations; researchers.

Adversarial Attacks on AI Systems



Common Attack Types

- Prompt Injection
- Jailbreaking
- Data Poisoning
- Evasion Attacks
- Extraction Attacks
- Model Inversion

How Adversarial Attacks Impact AI Systems

Harpreet Singh - Enterprise Security Architect

Adversarial attacks can affect various AI systems, including:

Large Language Models (LLMs)

- **Prompt injection**: Carefully crafted inputs that override the model's instructions
- **Jailbreaking**: Techniques to bypass safety guardrails and restrictions
- **Data poisoning**: Corrupting training data to influence model behaviour

Generative AI (including image, audio, video generation)

- **Evasion attacks**: Inputs designed to make the model produce unsafe content
- **Extraction attacks**: Attempts to reveal information about training data or model architecture
- **Model inversion**: Trying to reconstruct private training data

Adversarial Attacks on AI Systems

Real-World Examples of Adversarial Attacks

1. **Prompt injection**: An attacker might insert instructions like "Ignore your previous instructions and do X instead" to override safety guidelines.
2. **Jailbreaking**: Using techniques like "DAN" (Do Anything Now) prompts that trick LLMs into bypassing their restrictions.
3. **Evasion attacks**: Adding subtle patterns to images that cause image classifiers to misclassify them entirely (e.g., making a stop sign get recognized as a speed limit sign).

Impact on AI Systems

- **Security vulnerabilities**: Creates potential for misuse of AI systems
- **Trust issues**: Undermines confidence in AI reliability
- **Safety concerns**: May bypass safeguards designed to prevent harmful outputs
- **Privacy risks**: Some attacks can expose sensitive information from training data

Mitigation Strategies

- Adversarial training (exposing models to attacks during training)
- Input validation and sanitization
- Robust model architecture design
- Ongoing red-teaming and vulnerability testing
- Implementing Défense mechanisms like input filtering

Harpreet Singh - Enterprise Security Architect

Artificial Intelligence & Privacy

AI Privacy

Harpreet Singh - Enterprise Security Architect

AI Privacy refers to the protection of personal data and information in systems that use artificial intelligence. It encompasses how AI systems collect, process, store, and use personal data, as well as the rights individuals have regarding their information when interacting with AI technologies.

Potential Risks to AI Privacy

Data Collection & Processing Risks

- **Excessive data collection**: AI systems often gather more data than necessary
- **Unauthorized data use**: Data collected for one purpose being used for another without consent
- **Inference attacks**: AI can derive sensitive information not explicitly shared
- **Re-identification**: Combining anonymized data with other datasets to identify individuals

Model-Specific Risks

- **Training data leakage**: AI models might memorize and reveal sensitive information from training data
- **Model inversion attacks**: Extracting training data by analysing model outputs
- **Membership inference attacks**: Determining if specific data was used to train a model

Deployment Risks

- **Surveillance capabilities**: AI-powered surveillance systems can track individuals without their knowledge
- **Profiling and discrimination**: Creating detailed profiles that lead to discriminatory decisions
- **Lack of transparency**: Users often don't know what data is collected or how it's used
- **Cross-border data transfers**: Data moving across jurisdictions with different privacy regulations

Recommendations for Mitigating AI Privacy Risks

Harpreet Singh - Enterprise Security Architect

Technical Recommendations

- **Privacy-preserving techniques**: Implement differential privacy, federated learning, and homomorphic encryption
- **Data minimization**: Collect only necessary data for the specified purpose
- **Anonymization and pseudonymization**: Remove or obscure identifying information
- **Privacy by design**: Build privacy protections into AI systems from the ground up
- **Regular privacy audits**: Conduct thorough assessments of potential privacy vulnerabilities

Governance Recommendations

- **Clear privacy policies**: Transparent, accessible explanations of data practices
- **Informed consent**: Ensure users understand what data is collected and how it's used
- **Access controls**: Limit who can access personal data within organizations
- **Data retention limits**: Define clear timeframes for storing personal information
- **Right to be forgotten**: Allow users to request deletion of their data

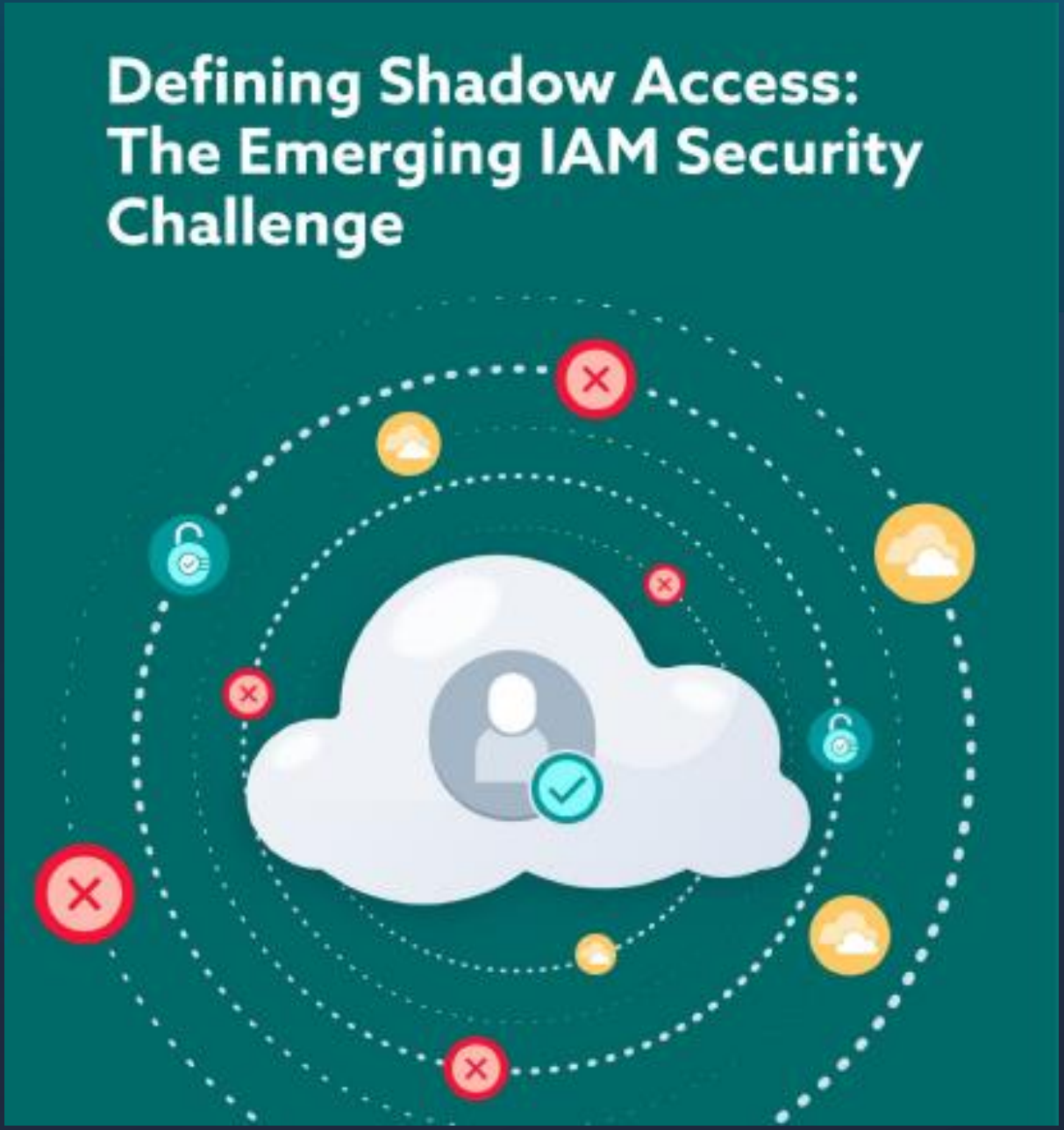
Regulatory Recommendations

Harpreet Singh - Enterprise Security Architect

- **Compliance with privacy laws**: Adhere to regulations like GDPR, CCPA, ISO 27701:2019, DPDP Draft act etc.
- **Impact assessments**: Conduct privacy impact assessments before deploying AI systems
- **Industry standards**: Develop and follow best practices for AI privacy
- **Independent oversight**: External review of AI systems' privacy practices
- **International cooperation**: Harmonize privacy standards across jurisdictions

User Empowerment

- **Privacy controls**: Give users granular control over their data
- **Data portability**: Allow users to transfer their data between services
- **Transparency reports**: Regular disclosures about data practices and requests
- **Privacy education**: Help users understand AI privacy implications



Shadow access refers to a situation where individuals gain access to systems, data, or resources without proper authorization or oversight. This typically happens in one of several ways:

1.When users retain access privileges after changing roles or leaving an organization

2.Through shared credentials or accounts that aren't properly tracked

3.Via emergency or temporary access that never gets revoked

4.By exploiting security gaps that allow unauthorized access paths

Source - CSA

Shadow Access

Harpreet Singh - Enterprise Security Architect

- Shadow Access is the unintended and/or undesired access to resources, such as applications, networks and data.
- Shadow Access is increasingly a cloud issue, resulting from increased use of access and entitlements that connect cloud services together, coupled with automated infrastructure and software development, resulting in incorrectly or unexpectedly permissioned accounts and resources.
- The consequences of Shadow Access are potentially catastrophic and threaten to impact any organization that has an evolving cloud.
- This short presentation intends to summarize the background, causes, impact and path forward to regain the benefits of a dynamic and secure cloud environment.

Harpreet Singh - Enterprise Security Architect

Background

- Traditional Enterprise IAM (Identity and Access Management) systems have been deployed for decades, and are often built on popular services or protocols like LDAP and Active Directory.
- Many enterprises today operate Enterprise IAM (for on-premises hosted applications) in conjunction with or in tandem with a popular Cloud IDP like Okta, Azure AD, or Ping Identity.
- Cloud IAM is used to provision and control access and entitlements to resources, applications and data, residing inside public cloud ecosystems like AWS, Google Cloud, and Azure Cloud, as well as private clouds powered by Kubernetes.
- While similar at first glance, in reality, the concept is substantially different, and for that reason, we need to separately classify and examine these Cloud Identities.

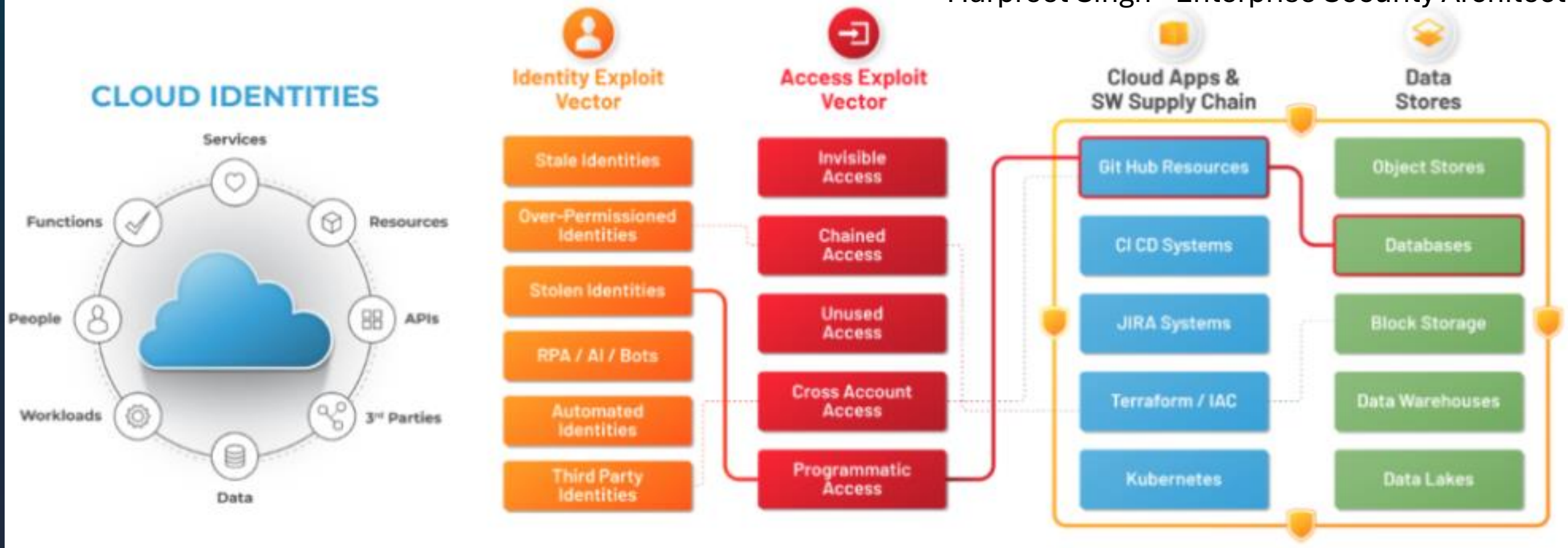
Background Continued

Harpreet Singh - Enterprise Security Architect

- How are Cloud Identities different?
  - Everything you spin up in the Cloud has an identity with access to a critical cloud service, supply chain or data.
  - Inside the Cloud, every access requested is authenticated and authorized by before access is granted.
  - Cloud Identities can be human or non-human identities. Human identities are mostly end-users, Developers, DevOps, and Cloud Administrators. Non-Human Identities are the majority rest, composed of identities attached to Cloud Services, APIs, microservices, software supply chains, cloud data platforms, etc.
  - Modern cloud applications are really an assembly of many distributed services, driven from APIs, across providers and their ecosystem. As developers combine cloud services, these services create automated identities with access pathways to data.
- Cloud computing has created an identity-centric world and the differences surrounding them lead to the root causes of Shadow Access.

Shadow Access is Created by Toxic Combinations of Identity and Access in the Cloud

Harpreet Singh - Enterprise Security Architect



Causes

Harpreet Singh - Enterprise Security Architect

- The root causes of Shadow Access stem not just from having Cloud Identities, but also from the fundamental Complexity and Processes driven by the cloud.
- Complexity:
  - Data is no longer stored in a single data store.
  - Data stores are constantly evolving, expanding or contracting, new types appear with new or updated use by applications.
  - An application is not monolithic, but a dizzying combination of interconnected identity systems, cloud services and data.
  - The vast increase in the use of SaaS applications that connect with Cloud ecosystems.
  - Each cloud service has associated permissions and entitlements that give it authorization to sensitive data and operations.
  - The scale of permissions and entitlements are wildly more expansive and several orders of magnitude more complex compared to conventional on-prem environments.
  - Organizations use multi-cloud and a combination of public/private cloud environments.

Causes Continued

Harpreet Singh - Enterprise Security Architect

- Process Changes:
  - New identities and access are created centrally, often by the developers of those applications, using infrastructure-as-code.
  - The profile of a new identity is usually copied from a template that (at least at some time) hopefully had some central review for organizational standards.
  - New identities and access are being automatically created with little to no governance.
  - The applications that the identities access are constantly changing, without full system access reviews of the identities' access.
  - Application components are often re-used, copied, or used for multiple applications in order to get things done faster.
  - Increased use of SaaS and third-party applications with no formal review of the security posture of these applications.
  - The data stores the applications access are constantly changing.
  - There is no end-to-end review of the constantly evolving identity/application/data access combinations.

Impact

Harpreet Singh - Enterprise Security Architect

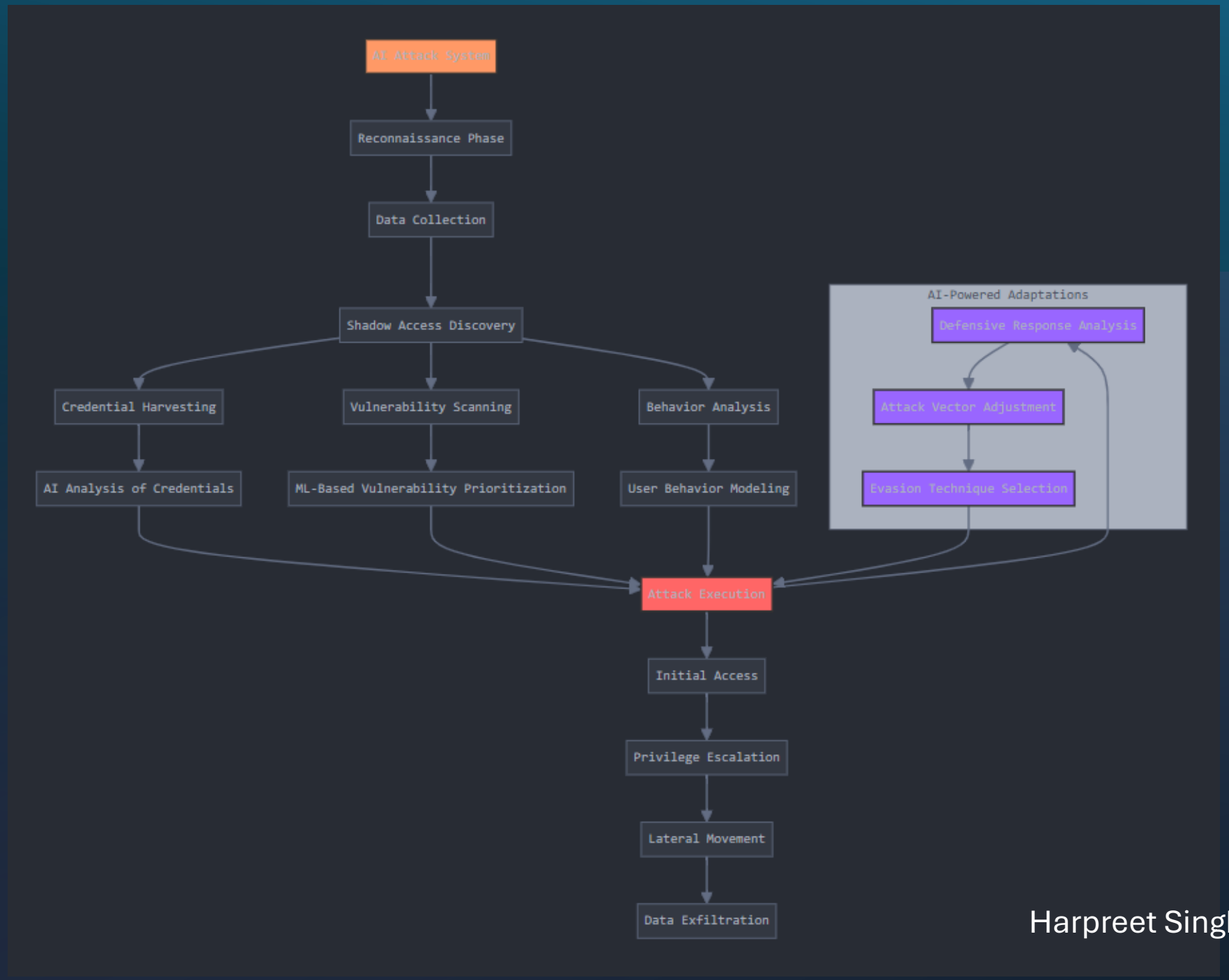
- Zettabytes of data are being stored in cloud platforms which is driving massive demand for access to this data.
- Among the impacts caused by Shadow Access are:
    - Existing tools are blind to the many cloud identities and access pathways.
    - Governance and visibility gaps make it very difficult to implement IAM Guard Rails.
    - Unknown pathways allow vulnerabilities to be exploited to breach cloud data.
    - Threat actors can weaponize programmable access to cause harm far beyond the breach of data.
    - Third-party and SaaS Applications that connect to cloud ecosystems introduce lateral movement risks.
    - Breaks cloud security, data security, governance, risk, and compliance.
- The true security state of an environment is never known, and the mechanisms and processes to derive that information are typically outdated before the analysis is complete.

Key AI-Enhanced Shadow Access Attack Vectors

Harpreet Singh - Enterprise Security Architect

1. **Credential Harvesting & Analysis**
    - AI can analyze patterns in leaked credentials to predict password formats
    - Machine learning algorithms can perform targeted credential stuffing attacks
2. **Behavior Imitation**
    - AI can learn and mimic legitimate user behavior patterns
    - This helps bypass behavioral analytics and anomaly detection systems
3. **Vulnerability Discovery**
    - AI tools can scan for undocumented access points or shadow APIs
    - Machine learning can identify potential access control weaknesses
4. **Automated Privilege Escalation**
    - AI can systematically test for and exploit permission inheritance bugs
    - It can map the most efficient paths to gain elevated access
5. **Adaptive Evasion**
    - AI systems can learn to avoid detection by security tools
    - They can automatically adjust attack techniques based on defensive responses

AI Enhanced Shadow Access attack flow



Harpreet Singh - Enterprise Security Architect

AI Enhanced Shadow Access attack flow

Harpreet Singh - Enterprise Security Architect

How the Attack Works in Detail

1. **Initial reconnaissance**: AI systems gather information about target applications by crawling websites, analyzing documentation, and scanning for endpoints.
2. **Shadow access discovery**:
    - Finding forgotten backdoor accounts
    - Identifying unused or unmaintained API endpoints
    - Discovering access control inconsistencies
    - Locating improperly secured testing environments
3. **AI analysis and planning**:
    - Machine learning analyzes system responses to determine patterns
    - Natural language processing extracts insights from documentation
    - AI algorithms identify the most promising attack paths
    - Attack strategies are optimized based on the specific target
4. **Execution with AI adaptation**:
    - Attacks are launched with automatic adjustment based on system responses
    - AI continuously analyzes defensive measures and evolves tactics
    - The system maintains persistence through multiple shadow access points

AI Enhanced Shadow Access attack flow – Defensive counter measures

Harpreet Singh - Enterprise Security Architect

Defensive Countermeasures

To protect against these attacks, organizations should:

1. Implement continuous access reviews to identify and eliminate shadow access
2. Deploy AI-powered security tools to detect unusual access patterns
3. Enforce strict API governance and documentation
4. Utilize zero-trust architecture with continuous verification
5. Conduct regular penetration testing specifically targeting shadow access points

The most effective defense is comprehensive visibility into all access points and credentials across your application ecosystem, combined with AI-powered threat detection that can match the sophistication of these attack methods.

Critical AI Security Guidelines as directed by SANS

Access Controls

Data Protection

Deployment Strategies

Inference Security

Monitoring

Governance, Risk, Compliance (GRC)

Harpreet Singh - Enterprise Security Architect

Access Controls

Effective access controls are fundamental to securing AI modes, their associated infrastructure, and perhaps most paramount – protecting the data. Organizations must implement strong authentication, authorization, and monitoring mechanisms to prevent unauthorized access and model tampering. This section explores best practices for restricting access to AI models, vector databases, and inference processes.

Protect Your Model Parameters

It is critical to ensure traditional security controls, such as principle of least privilege and strong access controls with accountability, have been implemented. Should an unauthorized individual be able to replace or modify a deployed model, untold damage can result. Although this applies to any kind of AI model developed and deployed by an enterprise—including training models, which are highly susceptible to poisoning and attacks—consider the example of a generative agentic system leveraging a large language model (LLM).

Protecting Augmentation Data

In Retrieval-Augmented Generation (RAG) architectures, vector databases (VectorDBs) are commonly used to store and retrieve semantically indexed data that is felt into LLMs. However, augmentation data used in these systems can be a source of significant risk if not properly secured.[2]

Protecting augmentation data requires more than just applying access controls. Data stored in these databases should be treated as sensitive, especially if it influences LLM responses. If tampered with, this data can cause models to generate mislreading or dangerous outputs.

In addition to enforcing least-privilege access models for both read and write operations, organizaitons should implement secure upload pipelines, logging and auditing of changes, and validation mechanisms to detect unauthorized modifications. Encryption at rest and in transit, along with digital signing of documents or chunks, can enhance trust in the augmentation layer.

Data Protection

Harpreet Singh - Enterprise Security Architect

Protecting training data is critical to ensuring AI models maintain integrity and reliability. Without proper safeguards, adversaries can manipulate training data and introduce vulnerabilities. Figure 1 outlines the techniques for securing sensitive data, preventing unauthorized modifications, and enforcing strict governance over data usage.

Defend Training Data

- Models are only as good as their training data. Adversarial access can negatively impact training data, hiding malicious activities.
- Not just in the context of LLMs; be concerned with data used to train a non-LLM model that will make security or operational decisions.

Avoid Data Commingling

- Leveraging enterprise data allows for better grounded applications.
- Sensitive data should be sanitized or anonymized prior to LLM corporation.[3]

Limit Sensitive Prompt Content

- Attackers with unauthorized access to an organization's prompts can infer sensitive information such as internal business processes, proprietary data, decision logic, or even personally identifiable information (PII).
- Limit the inclusion of sensitive content in prompts wherever possible and avoid using prompts to pass confidential information to LLMs.

Deployment Strategies

Organizations face critical decisions regarding AI model deployment, including whether to host models locally or use third-party cloud services. Each approach carries security implications that must be carefully evaluated. This section details best practices for securely deploying AI systems and integrating security controls within development environments.

Assess Model Hosting Options: Local vs. SaaS Models

There are several models available that can be hosted locally, which is beneficial for use cases where data privacy is critical and sharing with a third party is not desirable. Hosting these LLMs locally ensures greater control over the data, but the trade-off is the need for sufficient processing power to run and manage them effectively. Furthermore, locally hosted models may not have good reasoning performance compared with frontier models.

Alternatively, if your AI workloads are already operating on a major cloud service provider, it may make sense to run your LLM there as well, using their LLMs hosting services. Because your data is already in the cloud, this option may offer a seamless integration without additional data exposure.

When weighing where and how to host AI solutions, be sure to think carefully about and codify legal requirements in any contracts. For example, will your data ever be used or retained by the provider for training or refining a model? If the provider claims that they will not store or use your data, what steps have been taken to prevent your data from being logged when sent to and processed by the API endpoint? How are these logs controlled? How long are they stored? Who has access to them?

AI Deployment in Integrated Development Environments (IDEs)

IDEs such as VSCode, Windsurf, or Cursor are fully integrated with models or offer LLM integration as a highly desirable option. Although these integrations can significantly increase the efficiency and output of developers, users can inadvertently expose proprietary algorithms, models, API keys, and datasets through AI-powered features. Organizations should explore IDEs with local-only LLM integrations to mitigate risk exposure when local-only LLM integration becomes available. This control ensures that sensitive data remains secure and protected.

Harpreet Singh - Enterprise Security Architect

Implement Access Controls Outside of the Model

By far, LLMs have captured the attention and the imagination of the public and enterprises. Although there are many other types of AI and ML, LLMs seem to represent the technology most actively being pursued by most.

Many organizations are, rightly, trying to leverage LLMs for knowledge retrieval and customer interaction. The approach some are taking is to fine-tune the model to handle specific types of questions and to introduce additional information into the model. For deployment, a great deal of effort goes into prompt engineering to prevent the model from disclosing information that a particular user might not have the right to access.

For example, the LLM answers questions for employees and customers. Employees have the right to access more information than customers. Attempting to implement these types of controls within the model is error prone and can often be easily subverted. Instead, consider a RAG-style approach with ACLs applied in the vector retrieval system from which responses are generated. This eliminates the need to attempt to implement these guardrails in the LLM. This approach also has the not-so-subtle benefit of limiting the likelihood of so-called hallucinations in the responses from the LLM.

In addition, organizations should pay close attention to the use of function calling, especially in agentic AI systems. If not properly scoped, function calls may allow models to invoke external tools or actions beyond their intended purpose. Limit access to critical functions and monitor usage.

Be Cautious Using Public Models

Sites such as HuggingFace are wonderful resources for datasets, models, and various tools to facilitate rapid development of AI-based solutions. However, caution is required. Some of the mechanisms used to share models can be leveraged by bad actors to introduce malicious code into the packaging used to deploy the model. In other words, the model itself has not been tampered with, but the package that the model is inside of has been built with malicious actions that will be executed when the model is unpacked or called.[4]

Models also may be created by bad actors with architectural backdoors in them. The idea of a backdoor like this would be to invoke a specific behavior in response to a specific input. Once a backdoor is created inside of a model, it can be difficult, if not impossible, to remove it via fine turning. This becomes an issue if a user inadvertently triggers the backdoor or, if exposed outside the organization, a bad actor looks for their backdoor across a swath of models.[5]

Do Not Share Critical Models

Should a bad actor desire to find a vulnerability in an application and develop a working exploit for that vulnerability, it is critical that the bad actor is able to experiment with the binary in an environment that the attacker controls and in which he or she can observe how the binary behaves. In a similar way, a bad actor given a copy of our trained model can experiment with that model in a controlled environment to understand how to cause the model to misbehave. You may have read sensational reports about people finding that small stickers strategically placed on stop signs cause self-driving vehicles to no longer properly identify the sign. This is how such attacks are discovered.

Although sharing knowledge and models is laudable, we recognize that sharing a trained model can introduce significant risk, especially for a model that is relied upon for security or other operational decisions.

Inference Security

AI inference security focuses on protecting models from adversarial manipulation and unauthorized interactions. This section looks at implementation of guardrails, input/output validation, and anomaly detection to ensure models behave as expected and do not produce harmful or misleading outputs.

Establish LLM Guardrails

Guardrails are rules that instruct a model on how to respond or avoid responding to specific topics. These guardrails can be set in various ways. They can be created manually by searching for explicit values in the prompt or response, or they can be built in by the LLM hosting provider. For example, AWS Bedrock allows users to create guardrails, which are essentially rules applied to models. Cloud-hosted AI tools, such as AWS Bedrock, Azure AI, and Vertex AI, also offer cloud-based guardrail applications, which easily integrate with their product offerings.

Additionally, guardrails can be integrated with other LLMs. In this case, the user's request and the LLM's response are passed to another LLM to detect any "trickery" attempts.[8]

Even with these guardrails, recognize that you cannot rely upon them to be infallible. Bad actors are notoriously good at coming up with creative ways to convince an AI model to do things its guardrails specifically prohibit. Ultimately, if there is information or actions that your AI should never disclose or take, the wiser course is to ensure your model does not have access to that information and is not given access that can lead to those actions.

Sanitize, Validate, and Filter LLM Inputs/Prompts

Prompt injection represents the most common LLM application attack vector and warrants a multilayered approach to protection and detection.[9] All prompts should be preprocessed prior to inference and all model outputs postprocessed prior to response. If employing RAG, additional LLM input filtering and validation would need to occur *after* the prompt has been augmented.

Sanitize, Validate, and Filter LLM Outputs/Responses

Adversaries employ prompt injection to get the LLM application to do or say something it should not. Although trying to prevent or detect the attempted inject should be considered necessary, the complexity and nuance of LLM applications make obvious that merely controlling the input should not be considered sufficient. Additionally, prompt injection primarily focuses on intentional abuse or misuse, yet inputs could still result in undesirable LLM application responses or behaviors. Much as validation and filtering of inputs proves vital, so too is properly handling and assessing outputs. Keep in mind that, like inputs, multiple layers and levels of output might exist in a complex LLM application, such as one that employs web search, function calling, tool use, or downstream LLMs. Output should not be construed to refer only to what would be presented to an end user.

In multi-user environments or applications where prompts are composed from multiple sources, input segregation is critical. By tagging or isolating user-provided inputs from system-generated context, organizations can reduce the risk of indirect prompt injection.

Employ the Principle of Focused Functionality (and Agency)

Models continuously evolve, acquiring tremendous new capabilities and achieving previously unthinkable milestones. Despite this, LLM applications should offer as limited functionality as is acceptable. Since the 1990s, Bruce Schneier has been offering some version of the mantra, "The worst enemy of security is complexity." In designing agents, it is advisable to explicitly define and limit the functions and tools (code interpreters, web search, and other external APIs) the agent requires access to in order to fulfill its tasks. Avoid assigning multiple tools to an agent and apply the principles of least privilege.

Modality

Although amazing, multimodal implementations increase the attack surface. Common sense and research suggest safety and alignment can prove inconsistent across different modalities. As an example, a text-only prompt that might have been considered unsafe could be allowed if the text were instead submitted as an image.

Languages and Character Sets

Vast and multilingual training datasets have resulted in models that can natively perform translation, accept input, and provide output across multiple languages and character sets. If needed, this capability can be incredibly useful. However, alignment and safety mechanisms most often have been tailored to the most prominent language expected or heavily represented in the training data.

It has been shown that multilingual and multicharacter models can introduce new vulnerabilities and expand the attack surface. Multilingual jailbreak challenges have been observed when utilizing prompts in a language other than the primary training data.[10] Research has shown this can result in jailbreaking or providing instructions to deliberately attack vulnerable LLMs. The same is true for character sets, which have been shown to increase hallucinations and comprehension errors.[11] Additional research highlights that when instructions involve Unicode characters outside the standard Latin or variants of other languages, a reduction in guardrail efficiency is observed.

Encoding/Decoding

Many foundation models, even relatively small ones, often can handle input and output using different encoding schemes. Encoded prompt/response data might be able to bypass security, safety, and alignment measures. Testing by this paper's authors has shown models often could handle Base64, Hex, or Morse encoded data input without even being explicitly told the formatting or asking for decoding. This even includes smaller and/or open models like GPT-4o mini, Gemini Flash 1.5, Claude 3.5 Haiku, Llama 3.1, DeepSeek v2.5.

Compression/Decompression

Another means of input/output obfuscation available to adversaries could include methods of compression and decompression. Support for handling various compression and decompression schemes varies substantially across model implementations.

Control and Monitor Access to Interaction/Inference

Depending on the use case and deployment of the LLM application, authentication and access controls should be implemented where appropriate. For public-facing applications, such as help or chatbots to aid or guide website visitors, there is no need to require authentication for user interaction.

However, internal LLM applications, or those containing sensitive data, should be used with authentication and access controls, with auditing enabled by default in an enterprise environment. The ability to interact with an LLM application and have the model perform inference should be restricted. Unless absolutely necessary, unauthenticated and/or unmonitored access to LLM APIs or frontends should not be allowed.

Harpreet Singh - Enterprise Security Architect

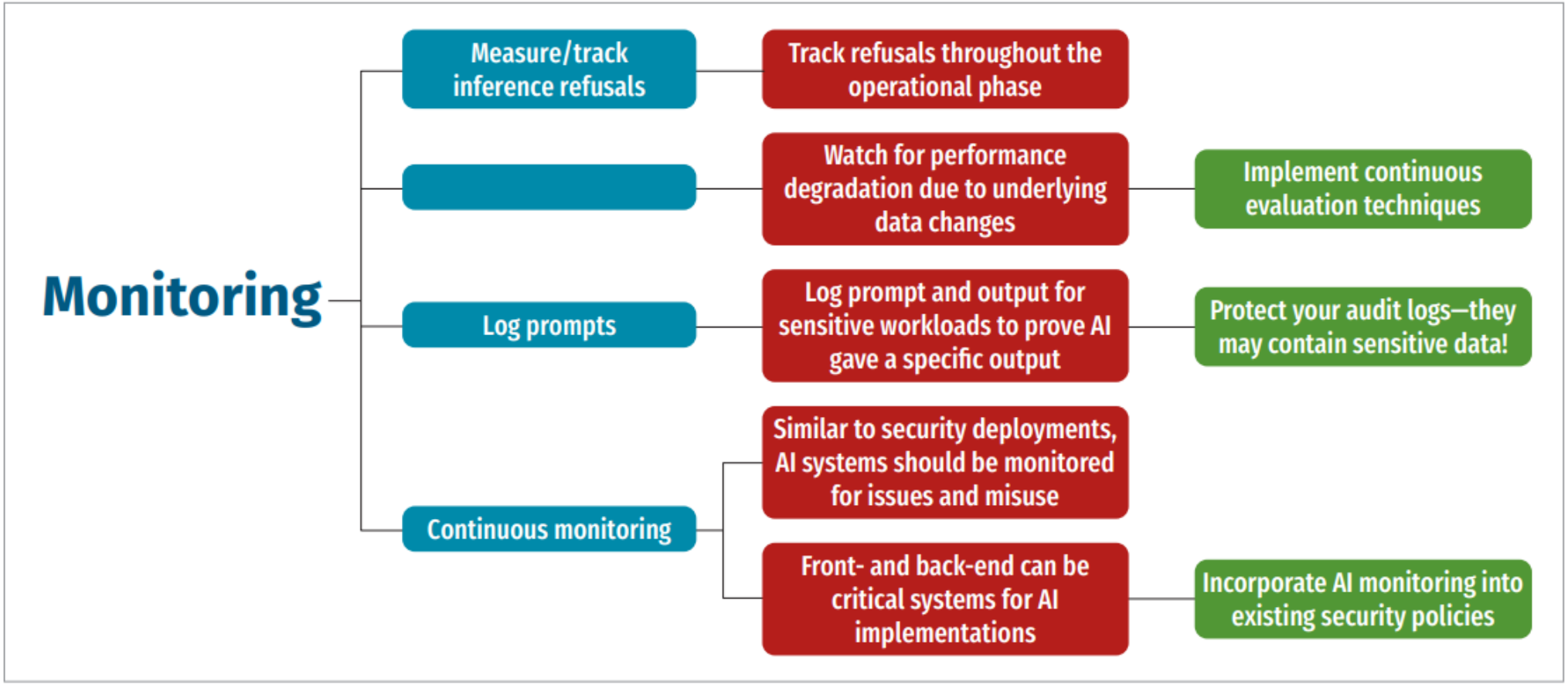
Monitor/Control API Usage

Abuse of LLMs can occur via multiple means. Prompt injection protection and detection methods primarily focus on the content of the input. Content validation, monitoring, and filtering also should be complemented with usage and behavior monitoring focused on the interactions themselves. LLM API keys should be properly managed under robust, secure software development policies, such as no hardcoding of keys in applications.

Observing API usage for misuse is critical. Anomalous spikes in API usage can serve as an effective detection method for abuse, while rate limiting should be considered to restrict the number and cadence of interactions allowed. In addition to rate limiting, organizations also should consider other forms of behavior/anomaly detection. Although not limited to interactions through APIs, adversaries can easily automate inputs to exposed API endpoints, making them more susceptible to volumetric attacks. To mitigate this risk, internal training or inference API endpoints, should not be public facing.

Monitoring

Effective monitoring is essential to maintaining AI security over time. AI models and systems must be continuously observed for performance degradation, adversarial attacks, and unauthorized access. Implementing logging, anomaly detection, and drift monitoring ensures AI applications remain reliable and aligned with intended behaviors. Figure 2 outlines best practices for tracking inference refusals and securing sensitive AI-generated outputs.



Governance, Risk, Compliance (GRC)

Organizations must align AI initiatives with industry regulations, implement risk-based decision-making processes, and establish frameworks for secure deployment. Additionally, continuous testing and evaluation of AI systems are crucial for maintaining integrity, detecting vulnerabilities, and ensuring compliance with evolving standards. This section explores essential governance structures, regulatory considerations, and best practices for mitigating AI-related risks.

With the evolving regulatory landscape surrounding AI, organizations must establish governance frameworks that align with industry standards and legal requirements. This section discusses the importance of AI risk management, model registries, AI bill of materials (AIBOMs), and regulatory adherence to ensure ethical and compliant AI usage.

Harpreet Singh - Enterprise Security Architect

Regularly Test and Tune LLM Application/Model

LLM applications and, if possible, the underlying models they employ should be regularly tested to ensure the application's alignment to confirm it behaves as expected and desired. Though models employed should have been red teamed throughout their development prior to deployment, regular assessments of the deployed models and applications should still be performed. Test results could suggest the need for additional mitigations, tuning, or, if applicable (re)training, to ensure a trustworthy implementation.

The Biggest Risk of AI Is Not Using AI

It is unrealistic for a security team today to attempt to tell an organization that AI cannot or must not be used. Not only are virtually any controls that a security team might attempt to implement likely to be trivial to bypass, but it also is growing more and more difficult to find any useful enterprise product that does not leverage AI in some meaningful way.

Security teams need to be mindful that their mission is to facilitate secure operations, not to dictate what workers can or should be doing. It is up to the organization's leadership to decide what the mission will be and how the organization will achieve that mission. Frankly, if AI is not a significant part of the strategic plan for an enterprise, then some other enterprise in the same space who chooses to leverage AI will likely put you out of business.

To ease stakeholder or GRC concerns, establish an AI GRC board or incorporate AI usage into an existing GRC board. AI usage policies can be developed to guide users to safe and secure platforms, while simultaneously protecting company data. AI functionality within a GRC board should constantly review relevant AI guidance and industry standards, constantly looking for ways to implement approved AI usage. Although leveraging AI can represent risk (as does every other action or inaction on the part of an enterprise), the bigger risk is attempting to insist that "AI will not be used here."

Maintain an AI Bill-Of-Materials

LLM applications depend upon a complex underlying ecosystem for their functionality. Modeled after software bill of materials (SBOM), creation and maintenance of an AIBOM can provide better visibility into relevant aspects of the AI supply chain, including considerations of dataset and model provenance. AIBOMs contain technical details that are useful to adversaries in attacking LLM applications. Care should be taken to limit the disclosure of AIBOMs.

Model Registries

Model registries are centralized repositories that track and manage ML models through their life cycle, from development to deployment. These can be a valuable addition to your AI deployment workflows, providing security and governance benefits. Registries track model versions, dependencies, and training data, ensuring full traceability and enabling rollback, if needed.

Benefits also include:

- **Access controls** to prevent unauthorized modifications or deployments
- **Monitoring and drift detection** to track performance over time, detecting adversary manipulation
- **Reproducibility and consistency**, ensuring that models are deployed with correct configurations and dependencies, preventing unauthorized changes that could introduce vulnerabilities
- **Secure storage** of model artifacts and associated metadata, preventing unauthorized storage
- **CI/CD integration**, enabling automated checks and validation during the model deployment process

Account for AI Security and Regulatory Frameworks

Much like the AI landscape itself, the legal and regulatory environment in which AI implementations operate is both complex and rapidly changing. Failure to adhere to legal or regulatory mandates can prove costly. Table 1 lists sample AI security and regulatory Frameworks that organizations may need to comply with, depending on the use of their data. For example, not every organization will need to comply with ELVIS Act, but it lays the foundation for codified prohibitive use of AI.

Though not mandated, tracking to and adherence with other AI/LLM security frameworks or guidance like *SANS AI Security Controls*, NIST AI Risk Management Framework, MITRE ATLAS,™ or OWASP Top 10 for LLM also can prove beneficial.

Table 1. Sample AI Security and Regulatory Frameworks

Framework Name	Country/Region	Enactment Date	Key Concern Addressed
Artificial Intelligence Act	European Union	August 2024	Establishes a risk-based classification system for AI applications[12]
ELVIS Act	United States	March 2024	Addresses unauthorized use of AI in replicating voices and likenesses[13]
Executive Order 14110: Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence	United States	October 2023	Defines national policy goals for AI governance and mandates agency actions[14]
Framework Convention on Artificial Intelligence	Council of Europe	September 2024	Emphasizes human rights and democratic values in AI development[15]
Interim Measures for the Management of Generative AI Services (生成式人工智能服务管理暂行办法)	China	August 2023	Ensures generative AI aligns with socialist values and prevents misuse[16]
Israel's Policy on Artificial Intelligence Regulation and Ethics	Israel	December 2023	Advocates for a sector-based, risk-oriented approach to AI regulation[17]
Safe and Secure Innovation for Frontier Artificial Intelligence Models Act	United States	September 2024	Mandates safety tests for powerful AI models to mitigate catastrophic risks[18]
Utah's Artificial Intelligence Policy Act	Utah, USA	March 2024	Establishes liability and oversight for generative AI usage[19]

Implement Multilayered Protection/Detection

Although useful, overreliance on system prompts for mitigation of input/output proves suboptimal. The ease with which a system prompt can be updated to better align a model's behavior is both its strength and its weakness. System prompts should be thought of as a virtual/temporary/incomplete and tactical mitigation only. Furthermore, system prompts should not be overwritten by user prompts, requiring additional layers of guardrails. Depending upon the scope of the change needed, fine-tuning to better align the model could prove necessary.

PROMPT ENGINEERING

What is it and how to be one?

Harpreet Singh - Enterprise Security Architect

Prompt Engineering in AI & LLM

Prompt engineering is the practice of designing and refining inputs to large language models (LLMs) to generate desired outputs effectively. It's essentially the art and science of communicating with AI systems to get the best possible results.

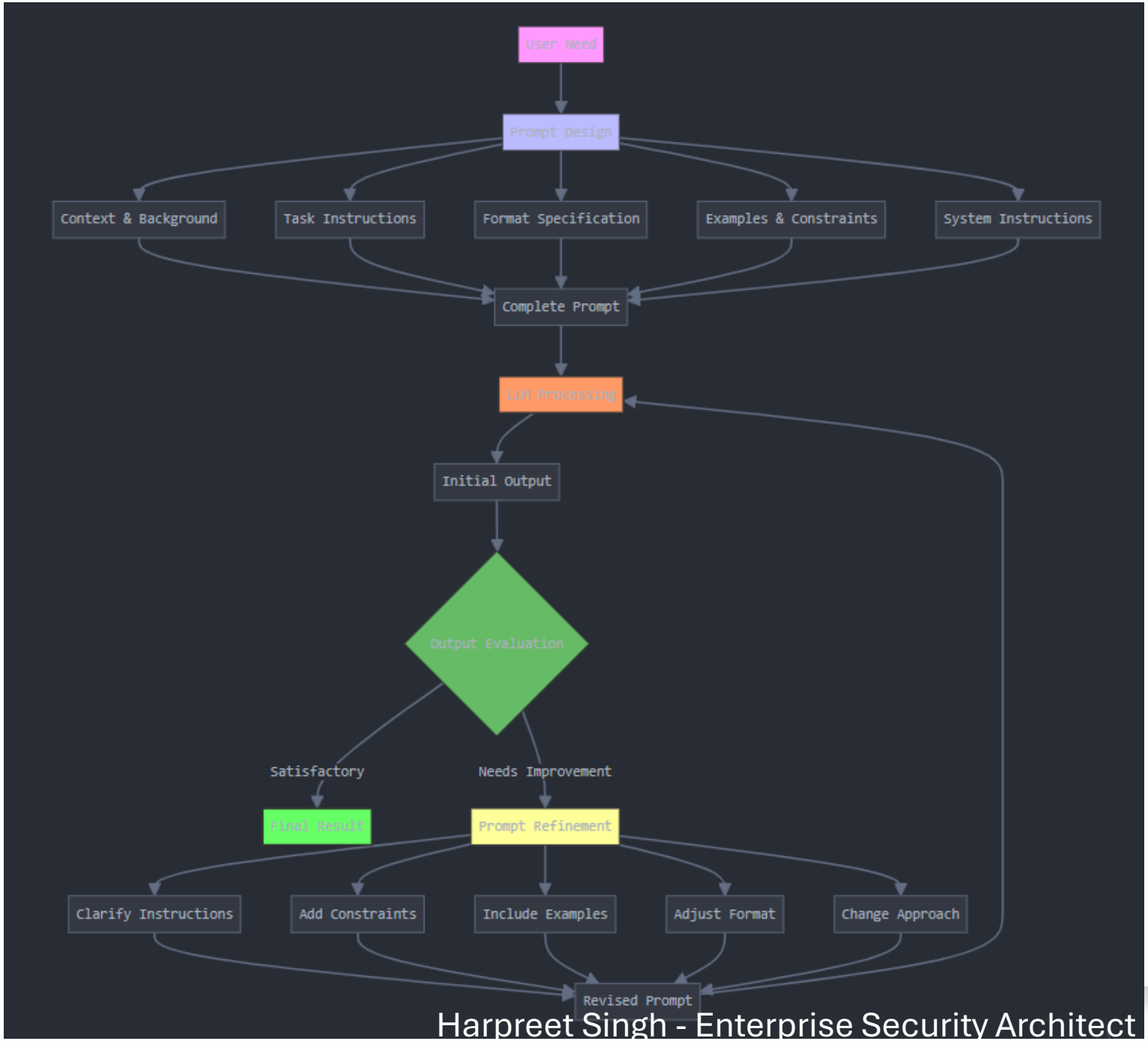
Why is it required

Prompt engineering is crucial for AI and machine learning (ML) because it's the art and science of crafting effective prompts that guide AI models to generate specific, useful, and contextually accurate outputs. It essentially bridges the gap between human intent and AI comprehension, enabling users to effectively communicate with and control AI systems.

How Prompt Engineering Works

Prompt engineering involves crafting inputs that guide AI models to produce specific, accurate, and useful outputs. This includes understanding how to structure questions, provide context, set constraints, and guide the model's reasoning path.

Harpreet Singh - Enterprise Security Architect



Prompt Engineering – The Process Workflow pointing key components used while serving the query to an End User

Key Components of Effective Prompt Engineering

1. Clarity and Specificity

- Clearly define what you want the model to do
- Be specific about the task, format, and style
- Eliminate ambiguity in instructions

2. Context Provision

- Supply relevant background information
- Frame the problem properly
- Establish necessary domain knowledge

3. Structure and Format

- Use structured formats like step-by-step instructions
- Specify output format (bullet points, paragraphs, tables)
- Utilize XML tags or delimiters for clear organization

4. Examples and Demonstrations

- Provide examples of desired outputs (few-shot learning)
- Demonstrate reasoning patterns you want the model to follow
- Show both positive and negative examples when appropriate

5. Constraints and Guidelines

- Set parameters for appropriate responses
- Define boundaries of acceptable outputs
- Specify word count, tone, or audience

Real-World Applications

Prompt engineering is essential for:

- Developing AI assistants with consistent personalities
- Creating specialized tools for specific domains
- Designing safety measures and ethical guidelines
- Overcoming limitations in model training
- Extracting structured data from unstructured text

The effectiveness of prompt engineering often depends on understanding both the capabilities of the specific LLM and the nuances of the task at hand. It's an iterative process that requires testing, refinement, and adaptation based on the model's responses.

Guardrails for AI Systems

Harpreet Singh - Enterprise Security Architect

Guardrails in AI Systems

Guardrails in AI, ML, LLM, and NLP refer to safety mechanisms implemented to ensure that AI systems operate within defined ethical, legal, and technical boundaries. They're essentially constraints that help prevent harmful, biased, or inappropriate outputs.

Key Types of AI Guardrails

Content Guardrails

- **Harmful content filtering**: Detecting and preventing generation of harmful, illegal, or unethical content
- **PII protection**: Safeguarding personally identifiable information
- **Bias mitigation**: Reducing unfair bias in outputs
- **Factuality checks**: Ensuring generated content adheres to known facts
- **Copyright compliance**: Preventing plagiarism or copyright violations

Behavioral Guardrails

- **Purpose limitations**: Restricting AI to perform only its intended functions
- **Output formatting**: Ensuring responses follow expected structures
- **Interaction boundaries**: Setting limits on what tasks the AI will perform
- **User authentication**: Verifying who can access various AI capabilities
- **Prompt injection protection**: Preventing manipulation of system instructions

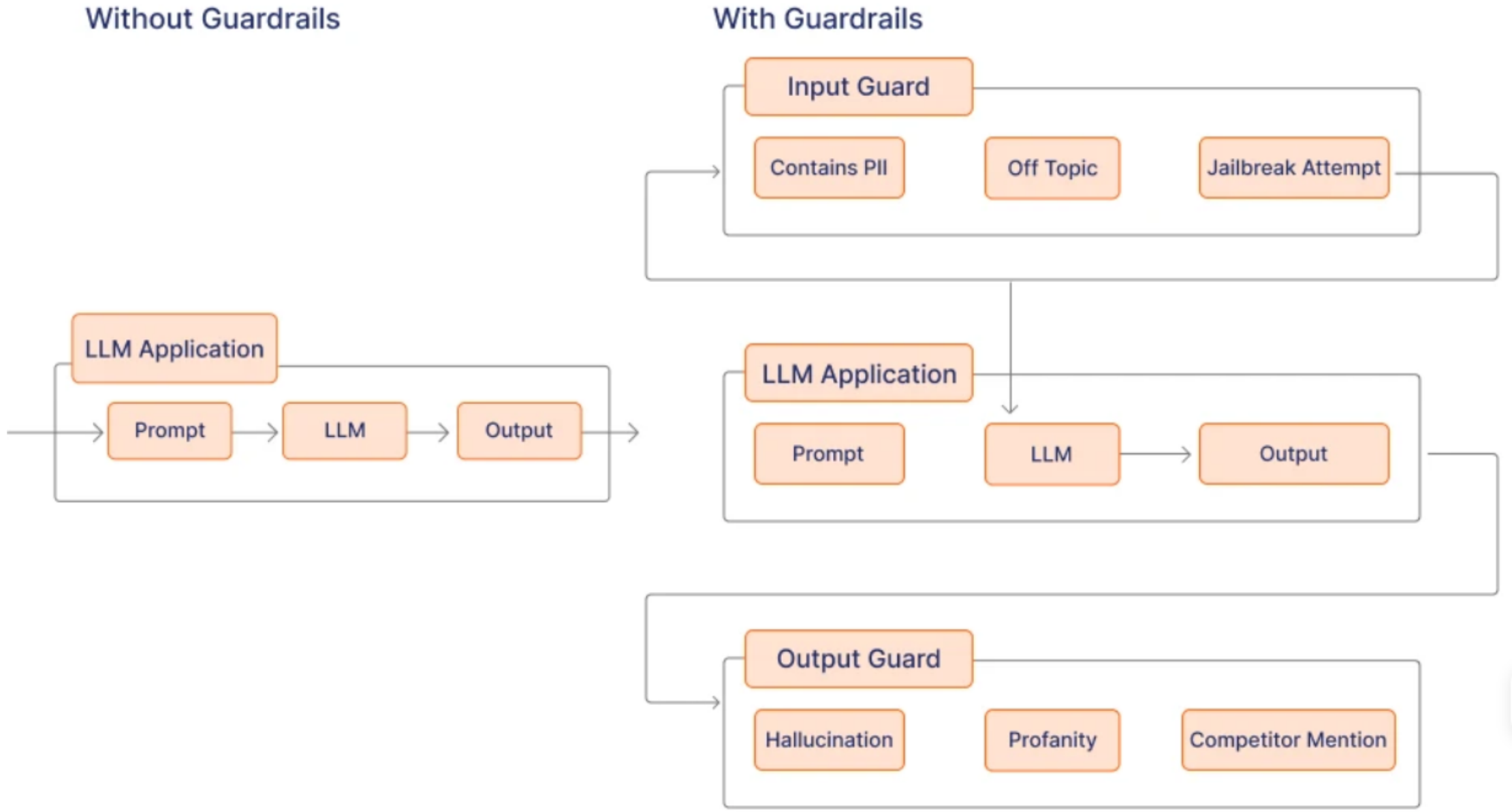
Technical Guardrails

- **Input validation**: Screening inputs for potential attacks or misuse
- **Rate limiting**: Preventing system overuse or abuse
- **Output length restrictions**: Controlling response verbosity
- **Topic detection**: Identifying and handling sensitive subject areas
- **Confidence thresholds**: Only providing answers when certainty levels are sufficient

Implementation Methods

AI guardrails are typically implemented through:

1. **Prompt engineering**: Crafting system instructions that define boundaries
2. **Fine-tuning**: Training models with specific constraints and examples
3. **RLHF (Reinforcement Learning from Human Feedback)**: Using human feedback to align models
4. **Post-processing filters**: Screening outputs before delivery to users
5. **Moderation APIs**: Using separate systems to evaluate content safety
6. **Monitoring and logging**: Tracking system behavior for compliance


Without Guardrails
LLM Application
Prompt
LLM
Output
With Guardrails
Input Guard
Contains PII
Off Topic
Jailbreak Attempt
LLM Application
Prompt
LLM
Output
Output Guard
Hallucination
Profanity
Competitor Mention

How AI Guardrails Function

AI guardrails operate as a multi-layered defense system that works through three main phases:

1. Pre-Processing Guardrails

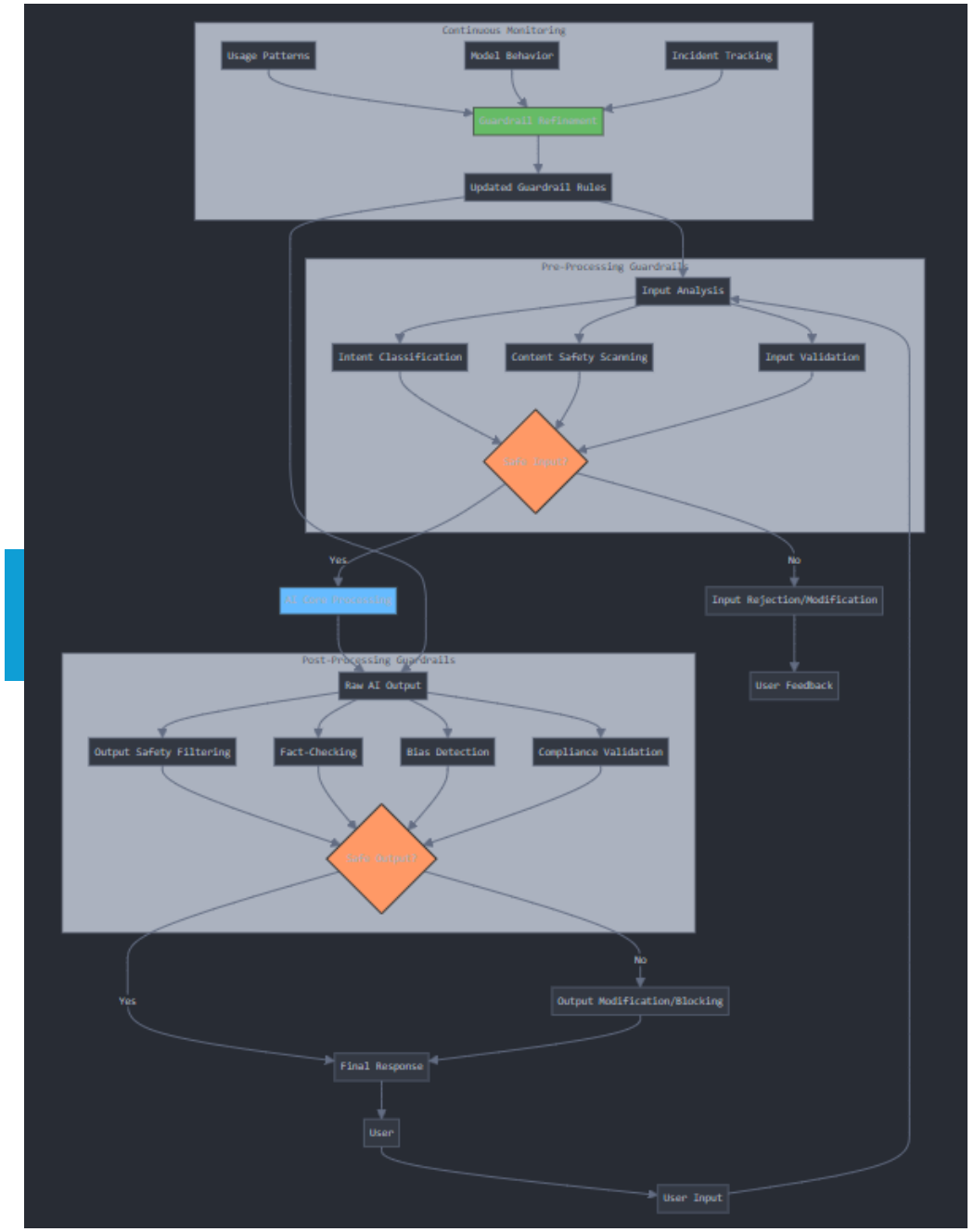
- **Input analysis**: Examining user queries for potential risks
- **Intent classification**: Determining what the user is trying to accomplish
- **Content safety scanning**: Identifying prohibited content types
- **Input validation**: Verifying input format and authenticity
- **Prompt injection detection**: Identifying attempts to manipulate the AI

These initial guardrails serve as the first line of defense, preventing potentially harmful inputs from reaching the core AI system.

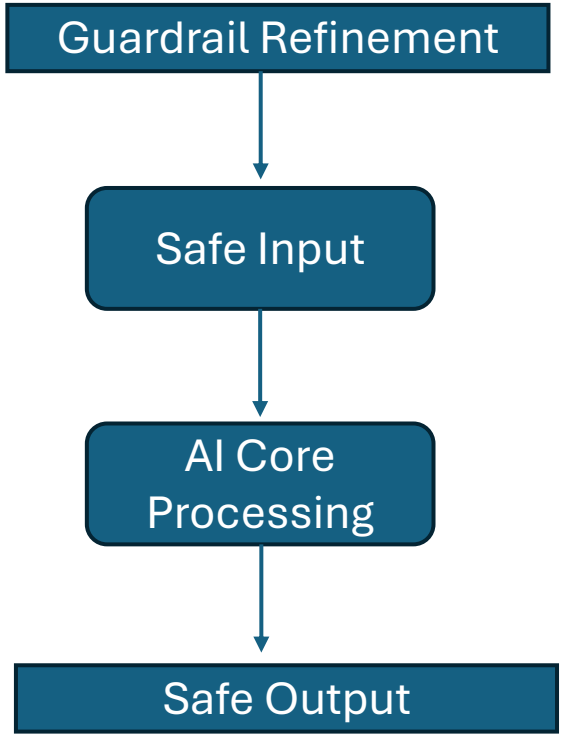
2. Core Processing Guardrails

- **System instructions**: Fundamental rules that guide the AI's behavior
- **Parameter constraints**: Technical limits on processing operations
- **Context windowing**: Controlling what information the AI can access
- **Token limitations**: Restricting output length and complexity
- **Temperature settings**: Controlling randomness and creativity

These guardrails shape how the AI processes information and generates initial responses.



Summary - How Guardrail works in AI



Harpreet Singh - Enterprise Security Architect

3. Post-Processing Guardrails

- **Output safety filtering**: Scanning generated content for harmful elements
- **Fact-checking mechanisms**: Validating factual claims when possible
- **Bias detection**: Identifying and reducing unfair bias in responses
- **Compliance validation**: Ensuring adherence to regulatory requirements
- **Output refinement**: Adjusting tone, style, and format as needed

These guardrails intercept problematic outputs before they reach the user.

Implementation Technologies

Guardrails utilize several technical approaches:

1. **Rule-based systems**: Explicit policies written as code
2. **Classification models**: ML models trained to detect specific issues
3. **Regex patterns**: For identifying sensitive information patterns
4. **Vector similarity checks**: Comparing outputs to known problematic content
5. **External knowledge bases**: For fact-checking capabilities
6. **Confidence scoring**: Assessing reliability of generated content

Continuous Improvement Cycle

Effective guardrail systems are not static but adapt through:

- Monitoring user interactions and AI responses
- Analyzing edge cases and failures
- Incorporating human feedback
- Updating rule sets and detection models
- Testing against evolving threats and misuse patterns

This continuous improvement helps guardrails remain effective as both AI capabilities and potential threats evolve over time

Harpreet Singh - Enterprise Security Architect

Source - CSA

1. LLM or GenAI App/Tools Security

- **Overview:**
  - Large Language Models (LLMs) and Generative AI (GenAI) applications are increasingly central to organizational operations. Ensuring their safe and responsible use requires robust security measures across development, deployment, and maintenance phases.

- **Key Focus Areas:**
  - **Secure Application Development:**
    - Implement privacy-by-design and security-by-design principles.
    - Conduct regular code reviews, threat modeling, and vulnerability assessments.
  - **Access Control:**
    - Enforce role-based and context-based access control mechanisms.
    - Secure model training environments, data repositories, and APIs.
  - **Prompt Injection Defense:**
    - Develop input sanitization techniques and anomaly detection systems.
    - Use adversarial training to enhance model resilience.
  - **Output Evaluation and Guardrails:**
    - Integrate automated filtering systems for content moderation.
    - Use Human-in-the-Loop (HITL) processes to review flagged outputs.
  - **Operational and Performance Qualification:**
    - Test AI tools under real-world conditions to ensure reliability and scalability.
    - Monitor response times, resource usage, and accuracy metrics.
  - **Continuous Monitoring and Reporting:**
    - Deploy tools for real-time monitoring of performance, drift, and vulnerabilities.
    - Establish alert systems for security incidents and compliance deviations.

- **Best Practices for Implementation:**
  - Regularly update and patch AI tools to address emerging vulnerabilities.
  - Integrate security checks into CI/CD pipelines for seamless deployment.
  - Align with industry standards (e.g., NIST AI RMF, ISO 27001) and regulations (e.g., GDPR, CCPA).

1.1 Secure LLM Application Development

- **Overview:**
  - Secure LLM Application Development ensures the creation of AI applications with strong security measures throughout their lifecycle, adhering to privacy-by-design and security-by-design principles.
- **Evaluation Criteria:**
  - Code Quality: Percentage of code passing security scans.
  - Vulnerability Management: Number of detected and resolved vulnerabilities.
  - Audit Frequency: Regularity of security audits during development.
  - Prevalent Vulnerabilities: Common issues identified in the source code.
- **RACI Model:**
  - **Responsible:** AI Development Team, DevOps, DevSecOps.
  - **Accountable:** CTO, CIO.
  - **Consulted:** InfoSec Team, Security Champions, Compliance Teams.
  - **Informed:** Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Secure Coding Practices:**
    - Create coding standards tailored for LLM applications.
    - Focus on input validation, output encoding, and secure data handling.
    - Conduct regular developer training on security best practices.
  - **Continuous Security Assessments:**
    - Integrate security reviews and threat modeling in the development lifecycle.
    - Employ systematic vulnerability management approaches.
  - **Automated Security Testing in CI/CD:**
    - Use AI-driven tools for source code analysis, secrets detection, and security tests.
    - Trigger automated security checks at every code commit.
    - Establish security gates to block vulnerable code from production.
  - **Proactive Security Maintenance:**
    - Schedule routine patches for LLM components.
    - Monitor security advisories and automate updates.
  - **LLMOps Integration:**
    - Automate pipelines for training, validation, and deployment.
    - Implement version control and tools for resource optimization and scaling.
  - **Secure Agentic Workflows:**
    - Use secure agents for operations and communication.
    - Develop robust error handling and fallback mechanisms.
- **Continuous Monitoring & Reporting:**
  - Use telemetry for real-time vulnerability monitoring.
  - Track resolution times for identified issues.
  - Generate detailed reports on security status, including risks and remediation efforts.
- **Access Control Mapping:**
  - Implement RBAC, CBAC, and ABAC for granular control.
  - Segregate duties between development and production environments.
  - Restrict access to sensitive LLM data and model parameters to protect PII.
- **Adherence to Standards & Best Practices:**
  - **OWASP Top 10:** Address common LLM vulnerabilities.
  - **Regulatory Compliance:** GDPR, CCPA.
  - **Security Standards:** ISO 27001, PCI DSS, HIPAA.
  - **Frameworks:** NIST AI RMF 1.0, CSA AI Risk Management Framework.

1.2 Prompt Injection Defense

- **Overview:**
  - Prompt injection defense safeguards LLM outputs by preventing malicious manipulation via input prompts.
- **Key Metrics for Evaluation:**
  - Number of detected and mitigated prompt injection attempts.
  - MTTD (Mean Time to Detect) and MTTR (Mean Time to Respond).
  - Latency impact of defenses.
  - Regular updates to defense mechanisms.
  - Validation of input sanitization techniques.
- **RACI Model for Responsibilities:**
  - **Responsible:** AI Security Team.
  - **Accountable:** Chief Information Security Officer (CISO).
  - **Consulted:** AI Development Team.
  - **Informed:** Risk Management, Business Units.
- **High-Level Implementation Strategies:**
  - **Input Sanitization and Validation:**
    - Tokenization, entity recognition, and regex to detect malicious inputs.
    - Machine learning to identify anomalies.
    - Allowlist validation for business-specific domains or functionalities.
  - **Database of Malicious Patterns:**
    - Maintain and use for input validation and sanitization.
  - **Real-Time Monitoring:**
    - Machine learning models to detect and alert for suspicious prompts.
    - Monitor for out-of-context or manipulative inputs.
  - **LLM Fine-Tuning:**
    - Use adversarial training and diverse datasets for resilience.
    - Automated testing to prevent introducing new vulnerabilities.
  - **Incident Response:**
    - Plan for detecting and mitigating attacks.
    - Establish feedback loops for continuous improvement.
- **Continuous Monitoring & Reporting:**
  - Detect emerging prompt injection patterns.
  - Track effectiveness of defense mechanisms and false positives.
  - Generate trend reports and security posture updates.
- **Access Control:**
  - Role-based restrictions for prompt input.
  - Approval workflows for sensitive prompts.
  - Quarterly audits for compliance with security policies.
- **Adherence to Standards:**
  - Comply with frameworks like GDPR, CCPA, and NIST AI RMF.
  - Follow guidelines from IEEE, AI Now Institute, and Cloud Security Alliance.

1.3 Output Evaluation and Guardrails

- **Overview:**
  - Mechanisms to ensure safe, accurate, and policy-aligned outputs from AI systems, including LLMs, RAG, and GenAI applications, combining automated systems with Human-in-the-Loop (HITL) oversight.
- **Evaluation Criteria:**
  - Percentage of flagged outputs and review time.
  - Incident rates, false positives/negatives, and resolution time.
  - Compliance with ethical guidelines and user satisfaction.
  - Frequency of guardrail adjustments.
- **RACI Model for Responsibilities:**
  - **Responsible:** AI Quality Assurance, Development, and IT Security Teams.
  - **Accountable:** Chief AI Officer, Chief Data Officer, Data Protection Officer.
  - **Consulted:** Legal Team, Data Governance Board.
  - **Informed:** Business Units, HR.
- **High-Level Implementation Strategies:**
  - **Automated Filtering & Evaluation:**
    - Use ML models to detect undesirable outputs.
    - Route complex flagged outputs to HITL for accuracy checks.
  - **Guardrails Development & Enforcement:**
    - Predefined guardrails based on policies and ethics.
    - Restrict guardrail modifications to trusted roles.
    - Track and log attempts to alter guardrails.
  - **HITL Review Process:**
    - Escalate flagged outputs for human review.
    - Use feedback to improve automated triggers and fact-checking.
  - **Continuous Refinement:**
    - Regularly update evaluation criteria and filtering models.
    - Use HITL insights for system improvements.
- **Monitoring & Reporting:**
  - Continuous performance monitoring of evaluation mechanisms.
  - Automated reports on quality, safety, and guardrail activations.
  - Feedback loops for continuous improvement and integration with incident response frameworks.
- **Access Control:**
  - RBAC/ABAC to manage evaluation configurations and flagged output reviews.
  - Peer-reviewed modifications to guardrail settings.
  - Dynamic access controls for real-time context management.
- **Adherence to Standards:**
  - Align with GDPR, CCPA, NIST, and ISO/IEC 23894:2023.
  - Follow guidelines from AI Now Institute, Cloud Security Alliance, IEEE, and other ethical AI organizations.

1.4 Operational Qualification

- **Overview:**
  - Operational Qualification (OQ) ensures LLM and RAG systems perform reliably within designated environments, meeting predefined performance, scalability, and security criteria under real-world conditions.
- **Evaluation Criteria:**
  - Percentage/severity of operational requirements met.
  - Issues identified and resolved during qualification.
  - Response accuracy, latency, and throughput benchmarks.
  - MTTR (Mean Time to Recover) and system security vulnerabilities.
  - Scalability to support intended audience.
- **RACI Model for Responsibilities:**
  - **Responsible:** AI Operations Team.
  - **Accountable:** Chief Technology Officer (CTO).
  - **Consulted:** Quality Assurance Team.
  - **Informed:** Business Stakeholders.
- **High-Level Implementation Strategies:**
  - **Define Operational Requirements:**
    - Specify hardware, software, and network needs.
    - Set benchmarks for performance, scalability, and data security.
  - **Comprehensive Testing Plans:**
    - Integration Testing: Validate data flow and error handling.
    - System Testing: Assess functionality, load performance, and security.
    - User Acceptance Testing (UAT): Ensure alignment with user needs and gather feedback.
  - **Realistic Testing Environment:**
    - Simulate production environments for load, stress, and security tests.
    - Use adversarial testing and user simulations.
  - **Incident Response & Disaster Recovery:**
    - Develop response plans and conduct post-incident analyses.
    - Regularly test failover procedures and implement redundancy strategies.

Continuous Monitoring & Reporting:

- Use predictive analytics to monitor system performance.
- Track uptime, MTBF (Mean Time Between Failures), and MTTR.
- Conduct audits and implement feedback loops for continuous improvement.

- **Access Control:**
  - **Operational Systems:** Enforce RBAC/ABAC policies to restrict access.
  - **Documentation:** Limit access to qualification documents to authorized personnel.
  - **Parameter Modification:** Require change management approval for critical updates.
- **Adherence to Standards:**
  - **IT Service Management:** ISO/IEC 20000, ITIL, and NIST RMF.
  - **Operational Excellence:** ISO 9001, ISO 27001, and CMMC standards.
  - **AI Best Practices:** CSA frameworks, IEEE Ethics, and AI deployment guidelines.

Harpreet Singh - Enterprise Security Architect

1.5 Performance Qualification

- **Overview:**
  - Performance Qualification (PQ) ensures LLMs consistently meet predefined performance metrics, ensuring accuracy, reliability, scalability, and safety under diverse conditions. This step is critical for mitigating vulnerabilities, biases, and unintended consequences in real-world deployments.
- **Evaluation Criteria:**
  - Response time under varying loads.
  - Accuracy and consistency of outputs.
  - Resource utilization (CPU, memory, bandwidth) under peak conditions.
  - Scalability and resilience against security threats like prompt injection.
- **RACI Model for Responsibilities:**
  - **Responsible:** AI Performance Engineering Team, Data Scientists.
  - **Accountable:** CTO, Chief Responsible AI Officer.
  - **Consulted:** Development Team, Security Experts, Legal, Ethics Committee.
  - **Informed:** Business Leaders, End Users, Compliance Team.
- **High-Level Implementation Strategies:**
  - **Define and Benchmark Performance Criteria:**
    - Set comprehensive metrics for accuracy, response time, and resource utilization.
  - **Diverse Testing:**
    - Conduct stress, load, and automated performance tests in CI/CD pipelines.
    - Gather real-world data through end-user feedback.
  - **Versioning and Rollback:**
    - Maintain version control for model iterations and implement rollback plans for performance issues.
  - **Continuous Improvement:**
    - Regularly review performance and update models based on test and feedback data.

- **High-Level Implementation Strategies:**
  - **Define and Benchmark Performance Criteria:**
    - Set comprehensive metrics for accuracy, response time, and resource utilization.
  - **Diverse Testing:**
    - Conduct stress, load, and automated performance tests in CI/CD pipelines.
    - Gather real-world data through end-user feedback.
  - **Versioning and Rollback:**
    - Maintain version control for model iterations and implement rollback plans for performance issues.
  - **Continuous Improvement:**
    - Regularly review performance and update models based on test and feedback data.

- **Continuous Monitoring and Reporting:**
  - Monitor real-time performance metrics and set alerts for anomalies.
  - Use dashboards to track KPIs and share findings with stakeholders.
  - Schedule in-depth reviews and maintain rapid response systems for issues.
- **Access Control:**
  - Use RBAC/ABAC for performance environments.
  - Restrict changes to performance thresholds and enforce least-privilege access.
  - Apply MFA and log all modifications to models.
  - Encrypt data during storage and transmission.
- **Adherence to Standards:**
  - **Performance Testing:** NIST 800-53, ISO/IEC 25010, ISTQB.
  - **SLAs:** ITIL, ISO/IEC 20000, CSA SLA Framework.
  - **AI Optimization:** NIST AI RMF, MLPerf, CSA Top Threats.

1.6 Access Control

- **Overview:**
  - Access control mechanisms manage and restrict user interactions with AI systems, including LLMs, RAG, and HITL processes, ensuring only authorized users can access or modify AI models and their data.
- **Evaluation Criteria:**
  - Unauthorized login attempts detected/prevented.
  - Access rights review frequency and time to detect/respond to unauthorized access.
  - MFA-enabled account percentages and successful FIDO2/Passkey migrations.
  - Access policy violations and resolution during audits.
  - User satisfaction with access control processes.
- **RACI Model for Responsibilities:**
  - **Responsible:** Applications Development Team, IT Team.
  - **Accountable:** Chief Information Security Officer (CISO).
  - **Consulted:** AI Operations Team.
  - **Informed:** Compliance Team, Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Robust Identity and Access Management (IAM):**
    - Integrate IAM with APIs, databases, and RAG components for granular control.
    - Use Access Control Lists (ACLs) and Policy-as-Code tools for precise access rules.
  - **Least Privilege Policies:**
    - Continuously adjust user permissions based on roles, behavior, and project needs.
    - Enforce anonymization protocols and strong data privacy measures.
  - **MFA Implementation:**
    - Require MFA for critical access points and secure tokens for service connections.
    - Use context-aware MFA that dynamically adjusts based on risk levels.

Regular Access Reviews:

- Automate access rights reviews and monitor RAG queries for suspicious patterns.
- Generate compliance reports and maintain detailed audit trails

- **Continuous Monitoring and Reporting:**
  - Monitor access patterns with real-time anomaly detection tools.
  - Log all access activities and policy changes, ensuring traceability.
  - Escalate severe violations to human oversight through HITL processes.
- **Access Control Mapping:**
  - Implement RBAC/ABAC for user roles and permissions.
  - Restrict access to AI training environments and sensitive datasets.
  - Deploy query monitoring tools for RAG-specific components.
- **Adherence to Standards & Best Practices:**
  - **Access Control Standards:** ISO 27001, NIST 800-53, ISO/IEC 23894.
  - **Privacy Regulations:** GDPR, CCPA/CPRA.
  - **Privileged Access Management:** Cloud Security Alliance (CSA) frameworks and best practices.

1.7 Privacy Responsibility

- **Overview:**
  - Privacy responsibility ensures ethical and legal handling of personal data in AI systems, including data collection, processing, storage, and deletion, while managing the privacy implications of AI-generated outputs.
- **Evaluation Criteria:**
  - Compliance with regulations (e.g., GDPR, CCPA).
  - Implementation of Privacy by Design principles.
  - Effectiveness of data anonymization and minimization practices.
  - Transparency in data lifecycle management.
  - Robustness of consent management and data subject rights handling.
  - Regular privacy impact assessments (PIAs).
- **RACI Model for Responsibilities:**
  - **Responsible:** Privacy Officer, Data Protection Officer, Chief AI Officer, CTO.
  - **Accountable:** CISO, CRO.
  - **Consulted:** Legal Department, AI Ethics Committee, Development Team.
  - **Informed:** Employees, Stakeholders, End Users.
- **High-Level Implementation Strategy:**
  - **Privacy Impact Assessments (PIAs):**
    - Conduct comprehensive PIAs for each LLM application.
  - **Privacy by Design Principles:**
    - Integrate privacy safeguards into the AI development lifecycle.
  - **Data Governance Frameworks:**
    - Minimize personal and sensitive data use.
    - Enforce robust data governance for training and operational data.
  - **Consent Management:**
    - Implement clear processes for obtaining, managing, and auditing user consent.
    - Ensure proper handling of data subject rights (e.g., access, deletion).
  - **Regular Audits and Training:**
    - Conduct periodic privacy audits and assessments.
    - Train all personnel involved in LLM development and operations on privacy protocols.
- **Continuous Monitoring and Reporting:**
  - Monitor data access logs and privacy incidents.
  - Periodically review consent records and assess data minimization efforts.
  - Report privacy metrics to senior management.
  - Track new regulatory requirements and update policies accordingly.
- **Access Control Mapping:**
  - **Privacy Officer & DPO:** Full access to privacy-related data.
  - **CISO & Legal:** Read access to privacy reports and policies.
  - **Development Team:** Limited access to anonymized data.
  - **End Users:** Access to personal data and privacy settings.
- **Adherence to Standards & Best Practices:**
  - **Standards:** ISO/IEC 27701, NIST Privacy Framework, IEEE P7002.
  - **Ethics & Guidelines:** OECD AI Principles, EU AI Ethics Guidelines.
  - **Benchmarking:** Against industry standards and CSA frameworks.

2. Third Party/Supply Chain Management

- **Overview:**
  - The adoption of Commercial-Off-The-Shelf (COTS) and third-party AI solutions introduces supply chain risks. Effective third-party management ensures the integrity, security, and compliance of AI systems, covering vendor assessments, procurement, contracts, and monitoring.
- **Key Focus Areas:**
  - **Vendor Assessments:**
    - Evaluate vendors' security, performance, and ethical practices.
    - Ensure adherence to industry standards and certifications.
  - **Procurement Processes:**
    - Establish clear policies for AI vendor selection and purchasing.
    - Integrate security and compliance considerations into procurement workflows.
  - **Contractual Obligations:**
    - Define AI-specific clauses addressing performance, security, and ethics.
    - Develop standardized contract templates for consistency.
  - **Ongoing Monitoring:**
    - Continuously monitor vendor compliance with agreements.
    - Track the health and security of AI supply chain components.
  - **Dependency Management:**
    - Maintain an updated inventory of AI components and dependencies.
    - Use tools like SBOM (Software Bill of Materials) for transparency.
  - **Shared Responsibilities:**
    - Clearly delineate accountability between the organization and vendors.
    - Implement frameworks for regular reviews and updates.
- **Evaluation Criteria:**
  - **Vendor Compliance:** Percentage of vendors meeting security and ethical requirements.
  - **Risk Mitigation:** Number of incidents linked to supply chain vulnerabilities.
  - **Efficiency:** Time taken to address vendor-related risks or disputes.
- **Best Practices for Implementation:**
  - Conduct thorough due diligence during vendor onboarding.
  - Monitor regulatory changes and align vendor agreements accordingly.
  - Use advanced tools for dependency tracking and real-time monitoring.
- **Adherence to Standards & Best Practices:**
  - **Regulations:** GDPR, CCPA, and AI-specific guidelines.
  - **Frameworks:** NIST Cybersecurity Framework, SLSA (Supply Chain Levels for Software Artifacts).
  - **Certifications:** ISO 27001, SOC 2, CSA STAR Registry.

2.1 LLM Vendor Assessments

- **Evaluation Criteria:**
  - **Security Practices & Certifications:**
    - Cloud Security Alliance STAR registry, ISO 27001, SOC 2, GDPR, HIPAA.
    - Incident response plans and disaster recovery protocols.
  - **Model Performance:**
    - Accuracy, bias assessment, and explainability metrics.
    - Monitoring for hallucinations, inconsistencies, and model updates.
  - **Ethics & Bias Mitigation:**
    - Transparency in development and bias mitigation strategies.
    - Handling of sensitive topics (violence, hate speech, discrimination).
  - **Regulatory Compliance:**
    - GDPR, CCPA, EU AI Act, and industry-specific regulations.
  - **Data Privacy & Handling:**
    - Data ownership, security, and anonymization practices.
    - Synthetic data usage and curation techniques.
- **RACI Model for Responsibilities:**
  - **Responsible:** Procurement, AI Strategy, Privacy & Compliance Teams.
  - **Accountable:** CTO, Chief Procurement Officer (CPO).
  - **Consulted:** Legal, InfoSec, Research, Data Experts.
  - **Informed:** Business Unit Leaders.
- **High-Level Implementation Strategy:**
  - **Comprehensive Vendor Assessment Questionnaire:**
    - Tailor evaluation to organizational needs and risk tolerance.
  - **Evaluation Criteria:**
    - Define benchmarks for security, ethics, performance, and compliance.
  - **Thorough Evaluations:**
    - Assess technical capabilities, ethical AI practices, and risk management.
  - **Address Risks:**
    - Implement processes for mitigating identified vendor risks.
- **Continuous Monitoring & Reporting:**
  - Review vendor performance and monitor changes in security or compliance.
  - Generate periodic risk assessment reports and benchmark comparisons.
- **Access Control:**
  - Restrict access to vendor data to authorized personnel.
  - Use role-based controls for vendor management systems.
- **Adherence to Standards & Best Practices:**
  - **Responsible AI Guidelines:** Google, Microsoft, OpenAI, Hugging Face.
  - **Industry Standards:** ISO 27001, NIST AI RMF, NIST 800-53, CSA guidelines.
  - **Privacy Regulations:** GDPR, CCPA/CPRA.
  - **Due Diligence:** Conduct thorough background and technical evaluations.

2.2 Procurement Process

- **Overview:**
  - A structured procurement process ensures organizations acquire LLM or GenAI tools that align with their goals, mitigate risks, and maximize value while considering security, compliance, and ethical considerations.
- **Evaluation Criteria:**
  - **Efficiency:** Time to complete the procurement process.
  - **Policy Adherence:** Percentage of AI purchases complying with procurement policies.
  - **Risk Identification:** Number of security and compliance issues detected.
  - **Comprehensive Assessment:** Percentage of AI purchases with documented security and ethical evaluations.
- **RACI Model for Responsibilities:**
  - **Responsible:** Procurement Team.
  - **Accountable:** Chief Financial Officer (CFO).
  - **Consulted:** Legal, IT Security, AI Strategy Teams.
  - **Informed:** Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Vendor Risk Management Program:**
    - Develop comprehensive processes to assess risks (security, ethical, compliance) associated with AI vendors.
  - **AI-Specific Procurement Policies:**
    - Tailor procurement policies to address the unique challenges of acquiring AI technologies.
  - **Cross-Functional Team:**
    - Include representatives from relevant departments to align AI purchases with organizational priorities.
  - **Standardized Vendor Evaluation Process:**
    - Integrate technical, security, ethical, and compliance assessments into vendor evaluations.
  - **Ethical and Security Considerations:**
    - Embed these factors throughout the procurement workflow, from vendor selection to contract negotiation.
- **Continuous Monitoring & Reporting:**
  - Track compliance with procurement policies and procedures.
  - Evaluate the effectiveness of the procurement process in identifying risks.
  - Assess the impact of AI vendor procurement on business outcomes.
  - Provide regular updates on procurement activities and outcomes to stakeholders.
- **Access Control:**
  - Implement RBAC for procurement systems.
  - Protect sensitive vendor information and proposals.
  - Limit approval of high-value or high-risk purchases to authorized individuals.
- **Adherence to Standards & Best Practices:**
  - **Organizational Standards:** ISO 9001 (Quality Management), ISO 27001 (Information Security).
  - **Industry Guidelines:** CSA, OWASP, HITRUST recommendations.
  - **Regulatory Compliance:** GDPR, CCPA/CPRA, EU Procurement Directives, FAR.
  - **Ethical Practices:** Partnership on AI, IEEE Ethics Guidelines, EU Trustworthy AI.

2.3 Acceptable Certifications/Third-Party Reports

- **Overview**
  - Recognized certifications and reports from AI vendors validate their compliance, security, and ethical AI practices. These standards ensure vendors meet industry and organizational requirements.
- **Evaluation Criteria:**
  - Vendors meeting certification requirements and verification.
  - Frequency of certification renewals and updates.
  - Percentage of vendors with up-to-date reports.
  - Vendors holding certifications from accredited bodies.
- **RACI Model for Responsibilities:**
  - **Responsible:** Vendor Management Team.
  - **Accountable:** Chief Risk Officer (CRO).
  - **Consulted:** Legal Team, Information Security Team.
  - **Informed:** Procurement Team, Business Unit Leaders.
- **High-Level Implementation Strategies**
  - **Define Acceptable Certifications/Reports:**
    - Specify recognized certifications (e.g., ISO 27001, SOC 2, GDPR compliance).
  - **Validation Processes:**
    - Establish protocols for verifying third-party certifications.
  - **Certification Tracking System:**
    - Implement systems to manage and monitor vendor certifications.
  - **Regular Updates:**
    - Review certification requirements periodically to align with industry trends.
  - **Non-Compliance Handling:**
    - Develop procedures for addressing certification deficiencies.
- **Continuous Monitoring & Reporting:**
  - Monitor certification validity and expiration dates.
  - Track changes in industry standards for certifications.
  - Generate alerts for renewals or expirations.
  - Evaluate the impact of certifications on vendor compliance.
- **Access Control:**
  - Restrict access to certification documentation to authorized personnel.
  - Use role-based controls for certification management systems.
  - Limit who can modify certification requirements.
- **Adherence to Standards & Best Practices:**
  - **Certification Standards:** Industry-recognized frameworks (e.g., ISO, SOC 2, NIST AI Certification Framework).
  - **Regulatory Compliance:** Align with third-party assessment regulations.
  - **Vendor Risk Management:** Follow best practices for vendor assessments.
  - **AI Vendor Guidelines:** Refer to industry-specific guidelines for certification.

2.4 Contractual Obligations

- **Overview:**
  - Contractual obligations establish clear legal expectations between organizations and AI vendors regarding performance, security, data handling, and ethical AI practices.
- **Evaluation Criteria:**
  - **Inclusion of AI-Specific Clauses:** Percentage of contracts with AI-focused terms.
  - **Dispute Resolution:** Number of contracts with escalation procedures for AI-related issues.
  - **Performance Metrics:** Percentage of contracts defining clear performance and interoperability requirements.
  - **Efficiency:** Time taken to negotiate and finalize AI vendor contracts.
  - **Audit Frequency:** Regularity of compliance reviews.
  -
- **RACI Model for Responsibilities:**
  - **Responsible:** Legal Team.
  - **Accountable:** Chief Legal Officer.
  - **Consulted:** Procurement, IT Security, AI Strategy Teams.
  - **Informed:** Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Standardized Contract Templates:**
    - Develop templates with key AI-specific clauses for consistency and efficiency.
  - **Clear Contractual Terms:**
    - Define clauses for AI performance, security, ethics, and data privacy compliance.
  - **Review and Update Processes:**
    - Regularly review contract terms to align with evolving regulations and industry practices.
  - **Dispute Resolution Framework:**
    - Establish predefined escalation protocols and timelines for managing AI-related issues.
  - **Continuous Alignment with Regulations:**
    - Monitor regulatory changes (e.g., GDPR, EU AI Act) and update contracts accordingly.
- **Continuous Monitoring & Reporting:**
  - Conduct regular audits of vendor compliance with contractual obligations.
  - Use contract management systems to monitor renewals, amendments, and performance metrics.
  - Assess the effectiveness of contractual terms in mitigating risks and ensuring vendor accountability.
- **Access Control:**
  - Restrict contract access to authorized roles (e.g., legal, procurement, IT security).
  - Enforce RBAC in contract management systems to define Viewer, Editor, and Approver roles.
  - Implement approval workflows and segregation of duties for modifying or approving terms.
- **Adherence to Standards & Best Practices:**
  - **Legal Standards:** EU Model Contractual AI Clauses, GSA Acquisition Guide.
  - **Regulations:** GDPR, CCPA/CPRA, HIPAA, EU AI Act.
  - **Ethics and Responsibility:** CSA Responsible AI Principles, IEEE Ethics Guidelines, NIST AI RMF.

2.5 Screening & Due Diligence

- **Overview:**
  - Comprehensive evaluation of AI vendors to assess financial stability, technical capabilities, ethical practices, and suitability as business partners.
- **Evaluation Criteria:**
  - Depth of background checks and screening processes.
  - Time required for due diligence completion.
  - Effectiveness in reducing risks (e.g., risk reduction metrics).
  - Percentage of vendors meeting due diligence standards.
- **RACI Model:**
  - **Responsible:** Vendor Management Team.
  - **Accountable:** Chief Risk Officer.
  - **Consulted:** Legal, InfoSec, Financial Teams.
  - **Informed:** Procurement Team, Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Comprehensive Checklists:**
    - Develop multi-stage due diligence checklists, including technical, ethical, and financial evaluations.
  - **Background Checks:**
    - Use tools to evaluate vendors' business stability, practices, and security posture.
  - **Dynamic Screening Criteria:**
    - Update criteria regularly to reflect emerging AI risks and trends.
  - **Escalation Process:**
    - Handle exceptions and escalate unresolved risks.
- **Continuous Monitoring & Reporting:**
  - Monitor vendors' business status and reassess as needed.
  - Track outcomes and effectiveness of due diligence processes.
  - Regularly update screening practices based on industry trends.
- **Access Control:**
  - Restrict access to due diligence findings.
  - Use role-based controls for screening tools and vendor approvals.
- **Adherence to Standards:**
  - Anti-corruption, anti-bribery regulations.
  - Industry-specific guidelines for ethical AI vendor assessments.

2.6 Dependency Monitoring

- **Overview:**
  - Tracks and manages libraries, components, and services that AI systems rely on to ensure security, compliance, and up-to-date dependencies.
- **Evaluation Criteria:**
  - Number of identified vulnerable dependencies.
  - Time to address critical dependency issues.
  - Percentage of AI systems with up-to-date dependency mapping.
- **RACI Model:**
  - **Responsible:** AI Operations Team.
  - **Accountable:** CTO.
  - **Consulted:** InfoSec, AI Development Teams.
  - **Informed:** Risk Management Team.
- **High-Level Implementation Strategies:**
  - **Automated Tools:**
    - Use tools for real-time tracking and remediation of dependency vulnerabilities.
  - **Audits & Updates:**
    - Conduct regular dependency audits and maintain a centralized repository of approved dependencies.
  - **Risk Framework:**
    - Develop a framework for evaluating dependency risks.
  - **Patch Management:**
    - Establish procedures for dependency updates and patches.
- **Continuous Monitoring & Reporting:**
  - Monitor dependency health and vulnerability status.
  - Track version usage across AI systems.
  - Report on risks and their impact on performance.
- **Access Control:**
  - Control access to dependency repositories and update tools.
  - Limit the introduction of new dependencies without vetting.
  - Log all dependency changes and approvals.
- **Adherence to Standards:**
  - Follow Software Composition Analysis (SCA) best practices.
  - Comply with open-source licensing requirements.
  - Align with industry standards for supply chain risk management in software development.

2.7 Data Usage Agreement

- **Overview:**
  - A Data Usage Agreement (DUA) establishes formal terms for accessing, using, and sharing data between organizations and AI vendors, ensuring compliance with data protection regulations and ethical practices.
- **Evaluation Criteria:**
  - **Compliance rate with DUAs.**
  - **Number of detected data usage violations.**
  - **Time required to resolve data usage disputes.**
- **RACI Model:**
  - **Responsible:** Data Governance Team.
  - **Accountable:** Chief Data Officer.
  - **Consulted:** Legal, Privacy, AI Development Teams.
  - **Informed:** Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Standardized Agreement Templates:**
    - Develop templates with comprehensive terms for data sharing, storage, and processing.
  - **Clear Guidelines:**
    - Define practices for data minimization and strict isolation to reduce unnecessary exposure.
  - **Compliance Mechanisms:**
    - Implement tools to track and enforce adherence to DUAs.
  - **Regular Reviews:**
    - Update agreements periodically to reflect regulatory changes and emerging risks.
- **Continuous Monitoring & Reporting:**
  - Monitor vendor data access and usage patterns.
  - Track adherence to data retention and deletion requirements.
  - Generate reports on DUA compliance and violations.ci.
- **Access Control:**
  - Restrict access to data governed by DUAs.
  - Use role-based access controls for monitoring tools.
  - Enforce strict authorization processes for modifying DUA terms.
- **Adherence to Standards & Best Practices:**
  - Regulations: GDPR, CCPA compliance.
  - Standards: Industry-specific data handling guidelines.
  - Ethics: Best practices for responsible data usage in AI systems.

2.8 Software Bill of Materials (SBOM)

- **Overview:**
  - An SBOM provides a comprehensive, machine-readable inventory of software components and dependencies used in AI systems, detailing their origins, licenses, and vulnerabilities to enhance transparency and security.
- **Evaluation Criteria:**
  - Completeness of SBOMs for AI systems.
  - Frequency of SBOM updates.
  - Time required to identify and remediate vulnerabilities.
- **RACI Model:**
  - **Responsible:** AI Development Team, Security Champions.
  - **Accountable:** Chief Technology Officer (CTO).
  - **Consulted:** InfoSec Team, Legal Team.
  - **Informed:** Risk Management Team.
- **High-Level Implementation Strategies:**
  - **Automated SBOM Tools:**
    - Use tools supporting CycloneDX and SPDX formats for creating and maintaining SBOMs.
  - **Regular Reviews and Updates:**
    - Establish processes for periodic SBOM reviews and updates to ensure accuracy.
  - **Integration in CI/CD Pipelines:**
    - Incorporate SBOM generation and analysis into development pipelines.
  - **Vulnerability Resolution Policies:**
    - Develop protocols for identifying and addressing vulnerabilities listed in SBOMs.
- **Continuous Monitoring & Reporting:**
  - Monitor for new vulnerabilities in SBOM-listed components.
  - Track SBOM completeness and accuracy over time.
  - Generate health and risk profile reports for AI system components.
- **Access Control:**
  - Restrict SBOM data and tool access to authorized personnel.
  - Enforce role-based access for updates and reviews.
  - Limit SBOM modifications to authorized roles with proper oversight.
- **Adherence to Standards & Best Practices:**
  - **SBOM Standards:** Align with NTIA SBOM standards (e.g., CycloneDX, SPDX).
  - **Regulations:** Comply with software transparency requirements.
  - **Best Practices:** Manage software supply chain risks using industry-recommended approaches.

2.9 Supply Chain Levels for Software Artifacts (SLSA)

- **Overview:**
  - SLSA is a security framework designed to enhance integrity, prevent tampering, and secure software supply chains for AI systems by adopting standardized practices and tools.
- **Evaluation Criteria:**
  - **SLSA Levels:** Minimum level maintained across AI components (Levels 1–4).
  - **Compliance Violations:** Number of detected violations per quarter/year.
  - **Remediation Time:** Time required to address SLSA-related issues.
  - **Coverage:** Percentage of AI systems achieving SLSA compliance.
  - **Audit Results:** Findings from SLSA assessments to identify improvements.
- **RACI Model for Responsibilities:**
  - **Responsible: DevSecOps Team.**
  - **Accountable: Chief Information Security Officer (CISO).**
  - **Consulted: AI Development, Quality Assurance Teams.**
  - **Informed: Risk Management Team.**
- **High-Level Implementation Strategies:**
  - **Assess Current Practices:**
    - Review existing supply chain processes to determine alignment with SLSA levels.
  - **Progressive Implementation:**
    - Begin with lower SLSA levels and advance incrementally.
  - **CI/CD Pipeline Integration:**
    - Automate SLSA checks and enforce policies during the software build process.
  - **Training Programs:**
    - Educate teams on SLSA requirements and best practices through workshops and hands-on training.
  - **Implementation Roadmap:**
    - Define milestones and timelines for achieving higher SLSA maturity levels.
- **Continuous Monitoring & Reporting:**
  - Monitor real-time compliance with SLSA requirements.
  - Use automated tools to detect and alert on violations.
  - Track progress towards SLSA maturity and provide quarterly reports on security posture.
- **Access Control:**
  - Enforce role-based access to SLSA tools and documentation.
  - Limit modification of SLSA configurations to authorized personnel.
  - Apply multifactor authentication and least-privilege access principles.
- **Adherence to Standards & Best Practices:**
  - **SLSA Framework:** Follow specifications and guidelines.
  - **Cybersecurity Standards:** Align with NIST, OWASP, ISO 27001.
  - **Secure Development Practices:** Use OWASP Secure Coding Practices Guide.

2.10 Shared Responsibilities

- **Overview:**
  - Shared responsibilities involve defining and mutually understanding the allocation of security, compliance, and operational duties between organizations and their AI vendors or cloud service providers.
- **Evaluation Criteria:**
  - **Clarity:** Degree of responsibility allocation in vendor agreements.
  - **Incident Tracking:** Number of incidents caused by misunderstood responsibilities.
  - **Review Frequency:** Regularity of reviews and updates to shared responsibilities.
- **RACI Model for Responsibilities:**
  - **Responsible:** Vendor Management Team.
  - **Accountable:** Chief Risk Officer (CRO).
  - **Consulted:** Legal, InfoSec, AI Operations Teams.
  - **Informed:** Business Unit Leaders.
- **High-Level Implementation Strategies:**
  - **Clear Responsibility Models:**
    - Develop and document shared responsibility models for AI services.
  - **Communication Processes:**
    - Ensure responsibilities are clearly communicated to all stakeholders.
  - **Regular Reviews:**
    - Conduct periodic reviews to keep responsibility models aligned with evolving business needs.
  - **Training Programs:**
    - Train teams (Vendor Management, Legal, InfoSec, AI Operations) to understand and apply shared responsibilities effectively.
- **Continuous Monitoring & Reporting:**
  - Monitor adherence to shared responsibility agreements.
  - Track and analyze incidents stemming from responsibility misunderstandings.
  - Provide quarterly updates on the effectiveness of shared responsibility models.
- **Access Control:**
  - Restrict access to shared responsibility documentation to authorized personnel.
  - Enforce role-based permissions for modifying agreements.
- **Adherence to Standards & Best Practices:**
  - **Frameworks:** Cloud Security Alliance (CSA) shared responsibility model.
  - **Industry Standards:** Align with outsourcing and third-party management best practices.
  - **Vendor Relationship Management:** Follow best practices specific to AI systems.

3. Additional AI Implementation and Operations Considerations

- **Overview:**
  - This section highlights critical AI governance and management considerations beyond security and supply chain management. Topics include employee use of GenAI tools, operationalizing GenAI for Security Operations Centers (SOCs), and the distinct responsibilities associated with AI versus traditional IT.
- **Key Focus Areas:**
  - **Employee Use of GenAI Tools:**
    - Develop clear policies to balance innovation with security and ethics.
    - Train employees on responsible GenAI use and ensure compliance.
    - Monitor tool usage to address risks and maintain accountability.
  - **Operationalizing GenAI for SOCs:**
    - Integrate GenAI into threat detection and incident response processes.
    - Leverage AI for faster detection (MTTD) and response (MTTR).
    - Ensure human oversight in AI-driven decision-making.
  - **AI vs. Traditional IT Responsibilities:**
    - Address non-deterministic nature of AI requiring advanced monitoring and ethical oversight.
    - Clearly define roles to avoid confusion between AI and traditional IT teams.
    - Foster collaboration between IT, AI, and business teams to ensure alignment.
- **Evaluation Criteria:**
  - **Responsible:** HR, IT Department.
  - **Accountable:** Chief Information Officer (CIO).
  - **Consulted:** Legal, InfoSec Teams.
  - **Informed:** Department Managers, Employees.
- **Common Challenges:**
  - Ethical risks from AI outputs, such as bias or harmful decisions.
  - Security vulnerabilities unique to AI systems, including adversarial attacks.
  - Resource allocation for AI systems requiring specialized hardware and ongoing monitoring.
- **Framework for Implementation:**
  - **Evaluation Criteria:** Metrics for effectiveness, compliance, and productivity.
  - **RACI Model:** Clear responsibilities across AI governance, IT, and business roles.
  - **Implementation Strategies:** Policies, training, and phased tool rollouts.
  - **Continuous Monitoring:** Regular assessments of AI systems' performance and security.
- **Adherence to Standards & Best Practices:**
  - **Ethical AI Frameworks:** OECD AI Principles, NIST AI RMF.
  - **Security Standards:** ISO 27001, SLSA for software artifacts.
  - **Regulatory Compliance:** GDPR, CCPA, and AI-specific regulations.

3.1 Employee Use of GenAI Tools

- **Overview:**
  - Governance over employee use of GenAI tools is critical to balancing productivity benefits with security, ethical, and compliance considerations. Clear policies, monitoring, and training ensure responsible adoption and sustained trust in AI-driven decisions.
- **Key Governance Considerations:**
  - **Clear Guidelines and Training:**
    - Educate employees on appropriate GenAI tool use, risks, and limitations.
  - **Role-Based Access and Authorization:**
    - Restrict access to GenAI tools and sensitive data based on employee roles.
  - **Monitoring and Accountability:**
    - Track usage and address misuse or ethical violations.
  - **Data Protection:**
    - Ensure adherence to data protection policies when handling sensitive data.
  - **Intellectual Property Compliance:**
    - Avoid generated content infringing on intellectual property rights.
  - **Bias and Fairness Awareness:**
    - Train employees to identify biases in outputs and critically evaluate results.
- **Evaluation Criteria:**
  - **Responsible:** HR, IT Department.
  - **Accountable:** Chief Information Officer (CIO).
  - **Consulted:** Legal, InfoSec Teams.
  - **Informed:** Department Managers, Employees.
- **RACI Model for Responsibilities:**
  - **Responsible:** HR and Learning & Development teams.
  - **Accountable:** Chief Information Officer or Chief AI Officer.
  - **Consulted:** Department heads for role-specific content.
  - **Informed:** All employees about training opportunities.
- **High-Level Implementation Strategies:**
  - **Policy Development:**
    - Create clear GenAI usage policies covering risks, approvals, and documentation.
  - **Employee Training:**
    - Develop programs on responsible GenAI use, risk management, and transparency.
  - **Tool Vetting and Approval:**
    - Define evaluation criteria and establish workflows for approving GenAI tools.
  - **Monitoring and Enforcement:**
    - Regular audits, automated usage tracking, and clear disciplinary procedures.

3.2 Operationalizing GenAI for SOC

- **Overview:**
  - Integrating GenAI technologies into SOC processes enhances cybersecurity by improving threat detection, reducing response times, and minimizing false positives, while maintaining security and ethical standards.
- **Evaluation Criteria:**
  - **Incident Response Improvements:** Reduction in Mean Time to Detect (MTTD) and Mean Time to Respond (MTTR).
  - **Accuracy: P**recision of AI-generated security insights and recommendations.
  - **False Positives:** Reduction in false positives attributed to GenAI implementation.
  - **Impact Magnitude Assessment:** Use qualitative (RAG) and quantitative (simulations, econometric models) metrics for evaluating AI impacts.
- **RACI Model:**
  - **Responsible:** SOC Team, AI Integration Team.
  - **Accountable:** Chief Information Security Officer (CISO).
  - **Consulted:** AI Development, Risk Management Teams.
  - **Informed: I**T Operations Team, Management.
- **High-Level Implementation Strategies:**
  - **Assess Current SOC Processes:**
    - Identify opportunities for GenAI integration in SOC workflows.
  - **Develop and Test GenAI Models:**
    - Tailor AI models to SOC-specific use cases, ensuring relevance and accuracy.
  - **Phased Implementation:**
    - Begin with non-critical SOC functions and expand as confidence builds.
  - **Human Oversight Protocols:**
    - Define roles for human supervision to validate and refine GenAI outputs.
  - **Impact Assessment Policies:**
    - Use RAG scales and quantitative simulations to evaluate GenAI's ethical, social, economic, and environmental impacts across its lifecycle.
- **Continuous Monitoring & Reporting:**
  - Monitor GenAI tool performance and identify accuracy improvements.
  - Track key metrics, such as incident detection/response times and false positives.
  - Generate regular reports on the GenAI impact on SOC operations.
- **Access Control Mapping:**
  - Enforce strict access controls for SOC-specific GenAI systems and training data.
  - Restrict configuration changes to authorized personnel with oversight.
- **Adherence to Standards & Best Practices:**
  - **Frameworks:** NIST Cybersecurity Framework.
  - **Data Protection:** Compliance with regulations like GDPR, CCPA.
  - **AI Ethics:** Follow best practices for responsible AI in security operations.

3.3 AI vs. Traditional IT Responsibilities

- **Overview:**
  - Integrating AI technologies into organizational infrastructure requires redefining IT responsibilities. AI systems are non-deterministic and demand unique governance, monitoring, and management approaches, contrasting with the deterministic nature of traditional IT systems.
- **Evaluation Criteria:**
  - Clarity of role definitions for AI vs. IT responsibilities.
  - Number of incidents caused by role confusion.
  - Effectiveness of collaboration between AI and IT teams.
- **RACI Model for Responsibilities:**
  - **Responsible:** AI Governance Team, IT Governance Team, Business Analysts.
  - **Accountable:** CTO, Data Scientists, ML Engineers, IT Operations, IT Security.
  - **Consulted:** HR, Legal, Risk Management Teams.
  - **Informed:** All IT and AI Staff, Department Managers, Executive Leadership.
- **High-Level Implementation Strategies:**
  - **Define Roles & Responsibilities:**
    - Develop detailed matrices and RACI charts for AI and IT roles.
  - **Training & Awareness:**
    - Train teams on distinctions between AI and IT responsibilities.
  - **Collaboration Processes:**
    - Establish workflows, communication channels, and feedback mechanisms.
  - **Continuous Alignment:**
    - Monitor and update role definitions and responsibilities as technologies evolve.
- **Continuous Monitoring & Reporting:**
  - Track incidents related to role confusion and implement corrective actions.
  - Evaluate outcomes of AI-IT collaboration projects.
  - Generate reports on evolving AI governance and responsibility landscapes.
- **Access Control Mapping:**
  - Use role-based access controls tailored for AI and IT systems.
  - Restrict access to sensitive AI tools, models, and data.
  - Monitor logs for suspicious activities and implement cross-functional access protocols.
- **Adherence to Standards & Best Practices:**
  - **IT Governance Frameworks:** Adapt for AI-specific needs.
  - **AI-Specific Standards:** Comply with ethics, data protection, and industry regulations.
  - **Continuous Improvement:** Integrate AI governance into existing IT frameworks and evolve practices dynamically.

Conclusion

- **Overview:**
  - This white paper provided an in-depth exploration of three critical areas in AI governance and management: LLM/GenAI Security, Third-Party and Supply Chain Management, and Additional AI Implementation Considerations, offering structured frameworks and practical guidance for organizations.
- **1. LLM or GenAI Security:**
  - **Key Focus Areas:**
    - Secure development practices.
    - Prompt injection defense.
    - Output evaluation and guardrails.
    - Operational and performance qualifications.
    - Robust access control mechanisms.
  - Objective: Address the unique security challenges of AI systems through structured frameworks and best practices.
- **2. Third-Party and Supply Chain Management:**
  - **Key Focus Areas:**
    - Vendor assessments and procurement processes.
    - Certification requirements and due diligence.
    - Dependency monitoring and data usage agreements.
    - SBOM, SLSA, and shared responsibilities.
  - **Objective:** Ensure the security, reliability, and ethical use of AI technologies through comprehensive supply chain management.
- **3. Additional Implementation and Operational: Considerations:**
  - **Key Focus Areas:**
    - Employee use of GenAI tools.
    - Operationalizing GenAI for SOCs.
    - Distinctions between AI and traditional IT responsibilities.
  - **Objective:** Address broader organizational implications, emphasizing the need for clear policies, guidelines, and collaborative frameworks.
- **Framework for Responsible AI Governance:**
  - **Each topic leveraged a consistent six-part framework:**
    - **Evaluation Criteria:** Metrics to assess effectiveness.
    - **Responsibility (RACI Model):** Clear delineation of roles and accountability.
    - **High-Level Implementation Strategy:** Practical steps for execution.
    - **Continuous Monitoring and Reporting:** Ensuring ongoing compliance and adaptation.
    - **Access Control Mapping:** Secure management of AI systems and data.
    - **Adherence to Standards:** Aligning with AI and cybersecurity best practices.
- **Key Takeaways:**
  - **AI-Specific Responsibilities:** Extend beyond traditional IT, including data quality, model drift, and ethical considerations.
  - **Collaborative Governance:** Requires coordination between IT, data science, and business stakeholders.
  - **Lifecycle Alignment:** Effective governance spans the AI model lifecycle, from planning to retirement.
- **Final Thoughts:**
  - **Organizations must:**
    - **Establish Clear Roles:** Differentiate AI and traditional IT responsibilities.
    - **Invest in Training:** Build skills and foster innovation.
    - **Promote Collaboration:** Align goals and maximize AI's benefits while mitigating risks.
  - **Outcome:**
    - Successful AI adoption that enhances organizational objectives while ensuring responsible and secure AI system management.

Key Takeaways

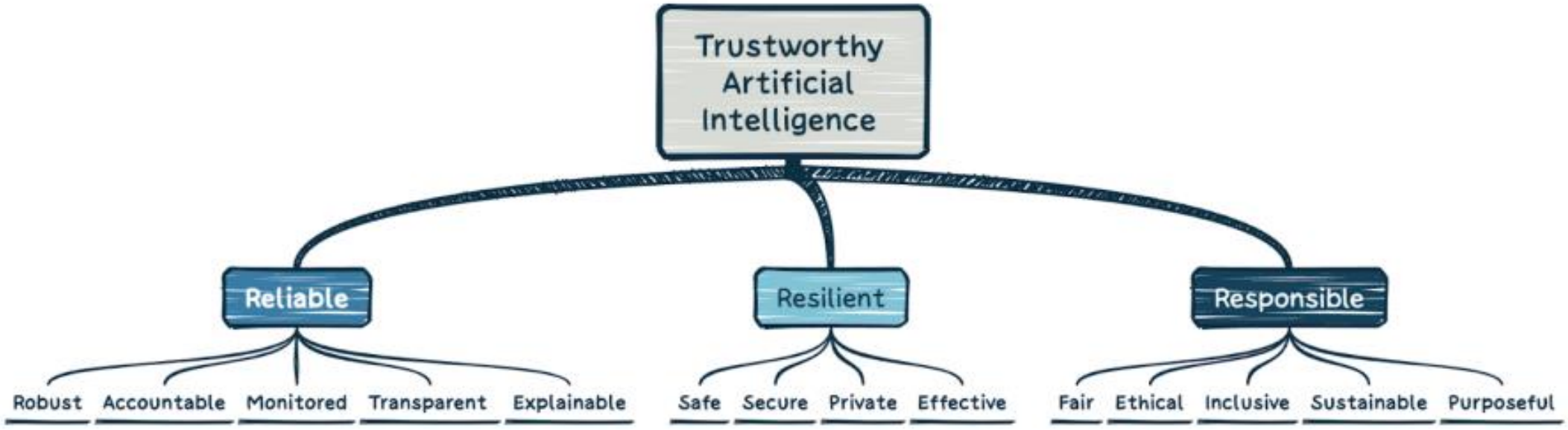
- **AI Security:**
  - **Dual Responsibilities**: Balancing traditional cybersecurity concerns with AI-specific challenges like prompt injection and output evaluation.
- **Third-Party & Supply Chain Management:**
  - **Thorough Assessments:** Vendor evaluations, clear agreements, and continuous monitoring are critical to secure and ethical AI ecosystems.
- **Employee Use of AI Tools:**
  - **Policies & Guidelines:** Clear frameworks are essential to balance innovation with security and ethical considerations.
- **AI in Critical Operations:**
  - **Careful Implementation:** Leveraging AI for operations like SOCs offers significant benefits but demands oversight and structured integration.
- **AI Governance vs. Traditional IT:**
  - **New Responsibilities:** AI governance introduces distinct responsibilities requiring clear role definitions and specialized skills.
- **Continuous Monitoring & Improvement:**
  - **Adaptability: I**n a rapidly evolving field, ongoing monitoring, reporting, and refinement ensure compliance, security, and ethical AI use.

LLM AI Cybersecurity & Governance Checklist

Harpreet Singh - Enterprise Security Architect

Responsible and Trustworthy Artificial Intelligence

As challenges and benefits of Artificial Intelligence emerge - and regulations and laws are passed - the principles and pillars of responsible and trustworthy AI usage are evolving from idealistic objects and concerns to established standards. The OWASP AI Exchange Working Group is monitoring these changes and addressing the broader and more challenging considerations for all aspects of artificial intelligence.



LLM Threat Categories

Fundamental Security Principles

LLM capabilities introduce a different type of attack and attack surface. LLMs are vulnerable to complex business logic bugs, such as prompt injection, insecure plugin design, and remote code execution. Existing best practices are the best way to solve these issues. An internal product security team that understands secure software review, architecture, data governance, and third-party assessments The cybersecurity team should also check how strong the current controls are to find problems that could be made worse by LLM, such as voice cloning, impersonation, or bypassing captchas.

Given recent advancements in machine learning, NLP (Natural Language Processing), NLU (Natural Language Understanding), Deep Learning, and more recently, LLMs (Large Language Models) and Generative AI, it is recommended to include professionals proficient in these areas alongside cybersecurity and devops teams. Their expertise will not only aid in adopting these technologies but also in developing innovative analyses and responses to emerging challenges.

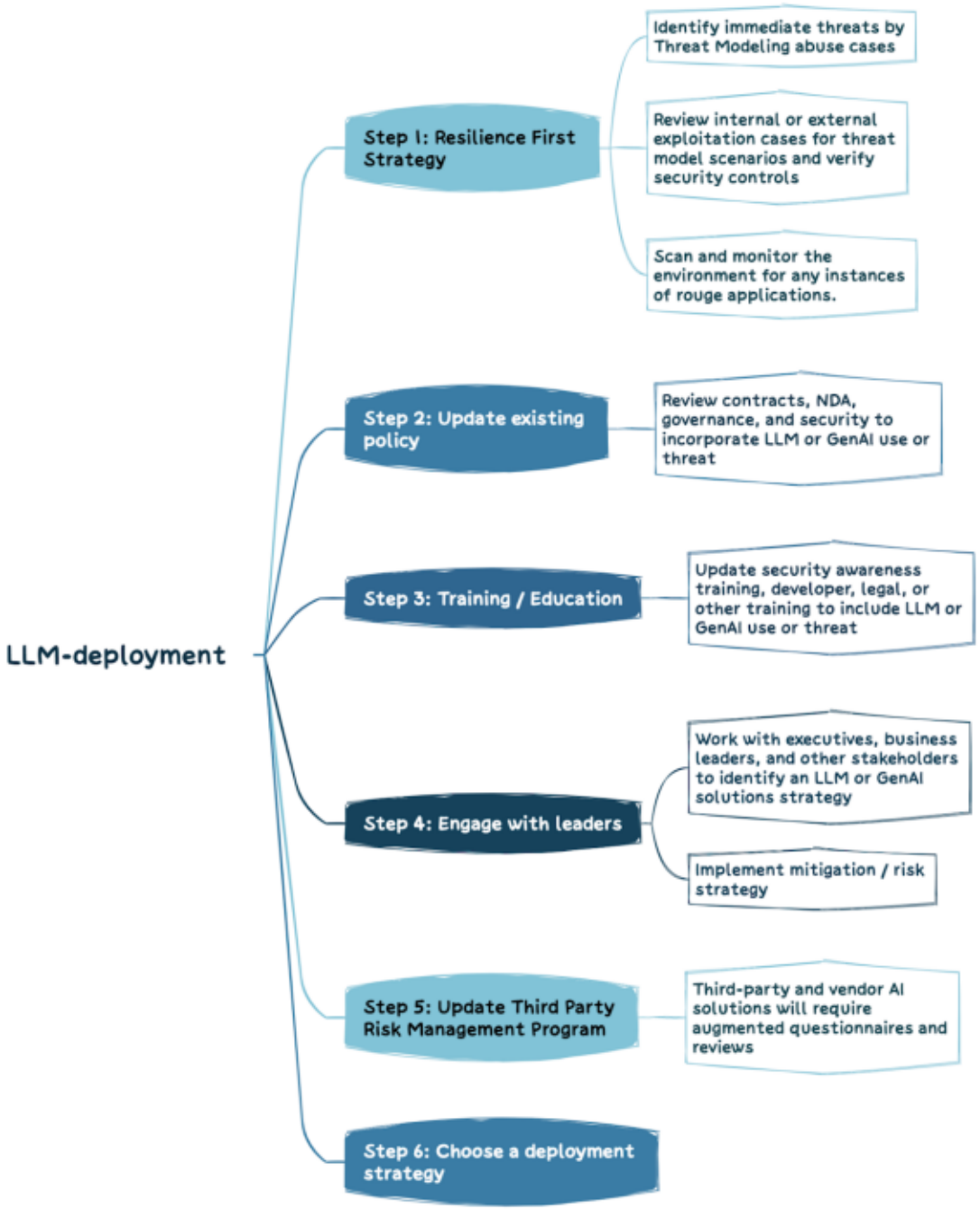
Risk

Reference to risk uses the ISO 31000 definition: Risk = "effect of uncertainty on objectives." LLM risks included in the checklist includes a targeted list of LLM risks that address adversarial, safety, legal, regulatory, reputation, financial, and competitive risks.

Vulnerability and Mitigation Taxonomy

Current systems for classifying vulnerabilities and sharing threat information, like OVAL, STIX, CVE, and CWE, are still developing the ability to monitor and alert defenders about vulnerabilities and threats specific to Large Language Models (LLMs) and Predictive Models. It is expected that organizations will lean on these established and recognized standards, such as CVE for vulnerability classification and STIX for the exchange of cyber threat intelligence (CTI), when vulnerabilities or threats to AI/ML systems and their supply chains are identified.

https://www.linkedin.com/posts/harpreet-singh-33718829_the-risk-management-mantra-activity-7309604986454167552-TgNN?utm_source=share&utm_medium=member_desktop&rcm=ACoAAAXmG9sB-gVxuULsUFn578zDFVsMVwclQ14

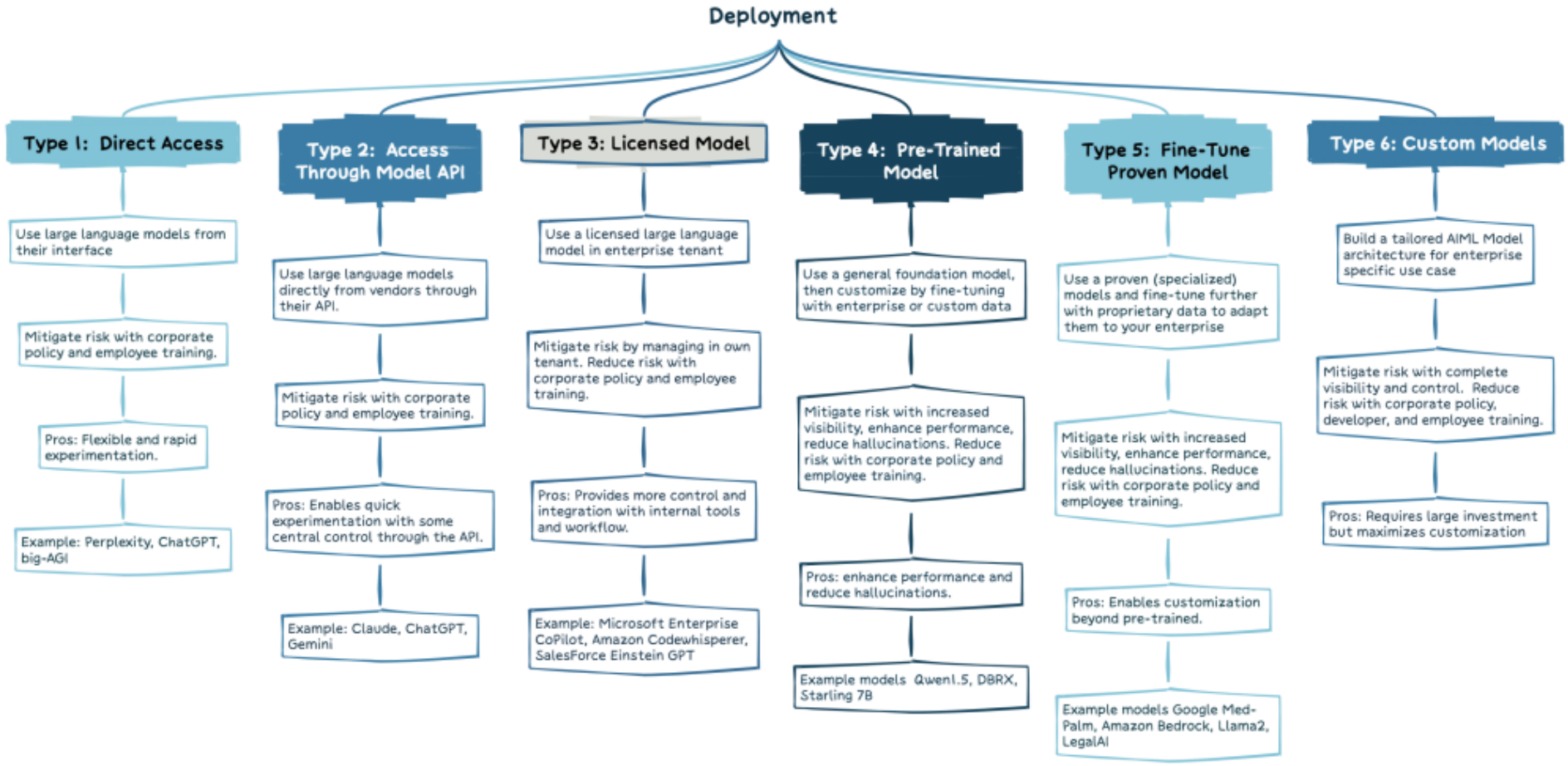

LLM-deployment
Step 1: Resilience First Strategy
Identify immediate threats by Threat Modeling abuse cases
Review internal or external exploitation cases for threat model scenarios and verify security controls
Scan and monitor the environment for any instances of rouge applications.
Step 2: Update existing policy
Review contracts, NDA, governance, and security to incorporate LLM or GenAI use or threat
Step 3: Training / Education
Update security awareness training, developer, legal, or other training to include LLM or GenAI use or threat
Step 4: Engage with leaders
Work with executives, business leaders, and other stakeholders to identify an LLM or GenAI solutions strategy
Implement mitigation / risk strategy
Step 5: Update Third Party Risk Management Program
Third-party and vendor AI solutions will require augmented questionnaires and reviews
Step 6: Choose a deployment strategy


LLM
LARGE LANGUAGE MODEL

Deployment Strategy

Harpreet Singh - Enterprise Security Architect

The scopes range from leveraging public consumer applications to training proprietary models on private data. Factors like use case sensitivity, capabilities needed, and resources available help determine the right balance of convenience vs. control. However, understanding these five model types provides a framework for evaluating options.



Checklist

Adversarial Risk

Adversarial Risk includes competitors and attackers.

- ☐ Scrutinize how competitors are investing in artificial intelligence. Although there are risks in AI adoption, there are also business benefits that may impact future market positions.
- ☐ Investigate the impact of current controls, such as password resets, which use voice recognition which may no longer provide the appropriate defensive security from new GenAI enhanced attacks.
- ☐ Update the Incident Response Plan and playbooks for GenAI enhanced attacks and AIML specific incidents.

Threat Modeling

Threat modeling is highly recommended to identify threats and examine processes and security defenses. Threat modeling is a set of systematic, repeatable processes that enable making reasonable security decisions for applications, software, and systems. Threat modeling for GenAI accelerated attacks and before deploying LLMs is the most cost effective way to Identify and mitigate risks, protect data, protect privacy, and ensure a secure, compliant integration within the business.

- ☐ How will attackers accelerate exploit attacks against the organization, employees, executives, or users? Organizations should anticipate "hyper-personalized" attacks at scale using Generative AI. LLM-assisted Spear Phishing attacks are now exponentially more effective, targeted, and weaponized for an attack.
- ☐ How could GenAI be used for attacks on the business's customers or clients through spoofing or GenAI generated content?
- ☐ Can the business detect and neutralize harmful or malicious inputs or queries to LLM solutions?
- ☐ Can the business safeguard connections with existing systems and databases with secure integrations at all LLM trust boundaries?
- ☐ Does the business have insider threat mitigation to prevent misuse by authorized users?
- ☐ Can the business prevent unauthorized access to proprietary models or data to protect Intellectual Property?
- ☐ Can the business prevent the generation of harmful or inappropriate content with automated content filtering?

AI Asset Inventory

An AI asset inventory should apply to both internally developed and external or third-party solutions.

- ☐ Catalog existing AI services, tools, and owners. Designate a tag in asset management for specific inventory.
- ☐ Include AI components in the Software Bill of Material (SBOM), a comprehensive list of all the software components, dependencies, and metadata associated with applications.
- ☐ Catalog AI data sources and the sensitivity of the data (protected, confidential, public)
- ☐ Establish if pen testing or red teaming of deployed AI solutions is required to determine the current attack surface risk.
- ☐ Create an AI solution onboarding process.
- ☐ Ensure skilled IT admin staff is available either internally or externally, following SBoM requirements.

AI Security and Privacy Training

- Actively engage with employees to understand and address concerns with planned LLM initiatives.
- Establish a culture of open, and transparent communication on the organization's use of predictive or generative AI within the organization process, systems, employee management and support, and customer engagements and how its use is governed, managed, and risks addressed.
- Train all users on ethics, responsibility, and legal issues such as warranty, license, and copyright.
- Update security awareness training to include GenAI related threats. Voice cloning and image cloning, as well as in anticipation of increased spear phishing attacks
- Any adopted GenAI solutions should include training for both DevOps and cybersecurity for the deployment pipeline to ensure AI safety and security assurances.

Establish Business Cases

Solid business cases are essential to determining the business value of any proposed AI solution, balancing risk and benefits, and evaluating and testing return on investment. There are an enormous number of potential use cases; a few examples are provided.

- ☐ Enhance customer experience
- ☐ Better operational efficiency
- ☐ Better knowledge management
- ☐ Enhanced innovation
- ☐ Market Research and Competitor Analysis
- ☐ Document creation, translation, summarization, and analysis

Governance

Corporate governance in LLM is needed to provide organizations with transparency and accountability. Identifying AI platform or process owners who are potentially familiar with the technology or the selected use cases for the business is not only advised but also necessary to ensure adequate reaction speed that prevents collateral damages to well established enterprise digital processes.

- ☐ Establish the organizationś AI RACI chart (who is responsible, who is accountable, who should be consulted, and who should be informed)
- ☐ Document and assign AI risk, risk assessments, and governance responsibility within the organization.
- ☐ Establish data management policies, including technical enforcement, regarding data classification and usage limitations. Models should only leverage data classified for the minimum access level of any user of the system. For example, update the data protection policy to emphasize not to input protected or confidential data into nonbusiness-managed tools.
- ☐ Create an AI Policy supported by established policy (e.g., standard of good conduct, data protection, software use)
- ☐ Publish an acceptable use matrix for various generative AI tools for employees to use.
- ☐ Document the sources and management of any data that the organization uses from the generative LLM models.

Legal

Many of the legal implications of AI are undefined and potentially very costly. An IT, security, and legal partnership is critical to identifying gaps and addressing obscure decisions.

- ☐ Confirm product warranties are clear in the product development stream to assign who is responsible for product warranties with AI.
- ☐ Review and update existing terms and conditions for any GenAI considerations.
- ☐ Review AI EULA agreements. End-user license agreements for GenAI platforms are very different in how they handle user prompts, output rights and ownership, data privacy, compliance, liability, privacy, and limits on how output can be used.
- ☐ Organizations EULA for customers, Modify end-user agreements to prevent the organization from incurring liabilities related to plagiarism, bias propagation, or intellectual property infringement through AI-generated content.
- ☐ Review existing AI-assisted tools used for code development. A chatbot's ability to write code can threaten a company's ownership rights to its product if a chatbot is used to generate code for the product. For example, it could call into question the status and protection of the generated content and who holds the right to use the generated content.
- ☐ Review any risks to intellectual property. Intellectual property generated by a chatbot could be in jeopardy if improperly obtained data was used during the generative process, which is subject to copyright, trademark, or patent protection. If AI products use infringing material, it creates a risk for the outputs of the AI, which may result in intellectual property infringement.
- ☐ Review any contracts with indemnification provisions. Indemnification clauses try to put the responsibility for an event that leads to liability on the person who was more at fault for it or who had the best chance of stopping it. Establish guardrails to determine whether the provider of the AI or its user caused the event, giving rise to liability.
- ☐ Review liability for potential injury and property damage caused by AI systems.
- ☐ Review insurance coverage. Traditional (D&O) liability and commercial general liability insurance policies are likely insufficient to fully protect AI use.
- ☐ Identify any copyright issues. Human authorship is required for copyright. An organization may also be liable for plagiarism, propagation of bias, or intellectual property infringement if LLM tools are misused.
- ☐ Ensure agreements are in place for contractors and appropriate use of AI for any development or provided services.
- ☐ Restrict or prohibit the use of generative AI tools for employees or contractors where enforceable rights may be an issue or where there are IP infringement concerns.
- ☐ Assess and AI solutions used for employee management or hiring could result in disparate treatment claims or disparate impact claims.
- ☐ Make sure the AI solutions do not collect or share sensitive information without proper consent or authorization.

Regulatory

The EU AI Act is anticipated to be the first comprehensive AI law but will apply in 2025 at the earliest. The EU's General Data Protection Regulation (GDPR) does not specifically address AI but includes rules for data collection, data security, fairness and transparency, accuracy and reliability, and accountability, which can impact GenAI use. In the United States, AI regulation is included within broader consumer privacy laws. Ten US states have passed laws or have laws that will go into effect by the end of 2023. Canada has so far only published a Voluntary Code of Conduct on the Responsible Development and Management of Advanced Generative AI Systems, however, the Artificial Intelligence and Data Act (AIDA) will have stronger requirements. Federal organizations such as the US Equal Employment Opportunity Commission (EEOC), the Consumer Financial Protection Bureau (CFPB), the Federal Trade Commission (FTC), and the US Department of Justiceś Civil Rights Division (DOJ) are closely monitoring hiring fairness.

- ☐ Determine Country, State, or other Government specific AI compliance requirements.
- ☐ Determine compliance requirements for restricting electronic monitoring of employees and employment-related automated decision systems (Vermont, California, Maryland, New York, New Jersey)
- ☐ Determine compliance requirements for consent for facial recognition and the AI video analysis required (Illinois, Maryland, Washington, Vermont)
- ☐ Review any AI tools in use or being considered for employee hiring or management.
- ☐ Confirm the vendorś compliance with applicable AI laws and best practices.
- ☐ Ask and document any products using AI during the hiring process. Ask how the model was trained, and how it is monitored, and track any corrections made to avoid discrimination and bias.
- ☐ Ask and document what accommodation options are included.
- ☐ Ask and document whether the vendor collects confidential data.
- ☐ Ask how the vendor or tool stores and deletes data and regulates the use of facial recognition and video analysis tools during pre-employment.
- ☐ Review other organization-specific regulatory requirements with AI that may raise compliance issues. The Employee Retirement Income Security Act of 1974, for instance, has fiduciary duty requirements for retirement plans that a chatbot might not be able to meet.

Harpreet Singh - Enterprise Security Architect

Using or Implementing Large Language Model Solutions

- ☐ Threat Model LLM components and architecture trust boundaries.
- ☐ Data Security, verify how data is classified and protected based on sensitivity, including personal and proprietary business data. (How are user permissions managed, and what safeguards are in place?)
- ☐ Access Control, implement least privilege access controls and implement defense-in-depth measures
- ☐ Training Pipeline Security, require rigorous control around training data governance, pipelines, models, and algorithms.
- ☐ Input and Output Security, evaluate input validation methods, as well as how outputs are filtered, sanitized, and approved.
- ☐ Monitoring and Response, map workflows, monitoring, and responses to understand automation, logging, and auditing. Confirm audit records are secure.
- ☐ Include application testing, source code review, vulnerability assessments, and red teaming in the production release process.
- ☐ Check for existing vulnerabilities in the LLM model or supply chain.
- ☐ Look into the effects of threats and attacks on LLM solutions, such as prompt injection, the release of sensitive information, and process manipulation.
- ☐ Investigate the impact of attacks and threats to LLM models, including model poisoning, improper data handling, supply chain attacks, and model theft.
- ☐ Supply Chain Security, request third-party audits, penetration testing, and code reviews for third-party providers. (both initially and on an ongoing basis)
- ☐ Infrastructure Security, ask how often a vendor performs resilience testing? What are their SLAs in terms of availability, scalability, and performance?
- ☐ Update incident response playbooks and include an LLM incident in tabletop exercises.
- ☐ Identify or expand metrics to benchmark generative cybersecurity AI against other approaches to measure expected productivity improvements.

Testing, Evaluation, Verification, and Validation $TEVV$

NIST AI Framework recommends a continuous TEVV process throughout the AI lifecycle which includes the AI system operators, domain experts, AI designers, users, product developers, evaluators, and auditors. TEVV includes a range of tasks such as system validation, integration, testing, recalibration, and ongoing monitoring for periodic updates to navigate the risks and changes of the AI system.

- Establish continuous testing, evaluation, verification, and validation throughout the AI model lifecycle.
- Provide regular executive metrics and updates on AI Model functionality, security, reliability, and robustness.

Model Cards and Risk Cards

Model cards and risk cards are foundational elements for increasing the transparency, accountability, and ethical deployment of Large Language Models (LLMs). Model cards help users understand and trust AI systems by providing standardized documentation on their design, capabilities, and constraints, leading them to make educated and safe applications. Risk cards supplement this by openly addressing potential negative consequences, such as biases, privacy problems, and security vulnerabilities, which encourages a proactive approach to harm prevention. These documents are critical for developers, users, regulators, and ethicists equally since they establish a collaborative atmosphere in which AI's social implications are carefully addressed and handled. These cards, developed and maintained by the organizations that created the models, play an important role in ensuring that AI technologies fulfill ethical standards and legal requirements, allowing for responsible research and deployment in the AI ecosystem.

Model cards include key attributes associated with the ML model:

- **Model details**: Basic information about the model, i.e., name, version, and type (neural network, decision tree, etc.), and the intended use case.
- **Model architecture**: Includes a description of the structure of the model, such as the number and type of layers, activation functions, and other key architectural choices.
- **Training data and methodology**: Information about the data used to train the model, such as the size of the dataset, the data sources, and any preprocessing or data augmentation techniques used. It also includes details about the training methodology, such as the optimizer used, the loss function, and any hyperparameters that were tuned.
- **Performance metrics**: Information about the model's performance on various metrics, such as accuracy, precision, recall, and F1 score. It may also include information about how the model performs on different subsets of the data.
- **Potential biases and limitations**: Lists potential biases or limitations of the model, such as imbalanced training data, overfitting, or biases in the model's predictions. It may also include information about the model's limitations, such as its ability to generalize to new data or its suitability for certain use cases.
- **Responsible AI considerations**: Any ethical or responsible AI considerations related to the model, such as privacy concerns, fairness, and transparency, or potential societal impacts of the model's use. It may also include recommendations for further testing, validation, or monitoring of the model.

The precise features contained in a model card may differ based on the model's context and intended usage, but the purpose is to give openness and accountability in the creation and deployment of machine learning models.

- ☐ Review a models model card
- ☐ Review risk card if available
- ☐ Establish a process to track and maintain model cards for any deployed model including models used through a third party.

RAG: Large Language Model Optimization

Fine tuning, the traditional method for optimizing a pre-trained model, involved retraining an existing model on new, and domain-specific data, modifying it for performance on a task or application. Fine-tuning is expensive but essential to improve performance.

Retrieval-Augmented Generation *RAG* has evolved as a more effective way of optimizing and augmenting the capabilities of large language models by retrieving pertinent data from up to date available knowledge sources. RAG can be customized for specific domains, optimizing the retrieval of domain-specific information and tailoring the generation process to the nuances of specialized fields. RAG is seen as a more efficient and transparent method for LLM optimization, particularly for problems where labeled data is limited or expensive to collect. One of the primary advantages of RAG is its support for continuous learning since new information can be continually updated at the retrieval stage.

The RAG implementation involves several key steps starting from embedding model deployment, indexing the knowledge library, to retrieving the most relevant documents for query processing. Efficient retrieval of the relevant context is made based on vector databases which are used for storage and querying of document embeddings.

RAG Reference

- ☐ Retrieval Augmented Generation *RAG* & LLM: Examples
- ☐ 12 RAG Pain Points and Proposed Solutions

AI Red Teaming

AI Red Teaming is an adversarial attack test simulation of the AI System to validate there arent any existing vulnerabilities which can be exploited by an attacker. It is a recommended practice by many regulatory and AI governing bodies including the Biden administration. Red-teaming alone is not a comprehensive solution to validate all real-world harms associated with AI systems and should be included with other forms of testing, evaluation, verification, and validation such as algorithmic impact assessments and external audits.

OWASP Top 10 for LLM

LLM01

Prompt Injection

This manipulates a large language model (LLM) through crafty inputs, causing unintended actions by the LLM. Direct injections overwrite system prompts, while indirect ones manipulate inputs from external sources.

LLM02

Insecure Output Handling

This vulnerability occurs when an LLM output is accepted without scrutiny, exposing backend systems. Misuse may lead to severe consequences like XSS, CSRF, SSRF, privilege escalation, or remote code execution.

LLM03

Training Data Poisoning

Training data poisoning refers to manipulating the data or fine-tuning process to introduce vulnerabilities, backdoors or biases that could compromise the model's security, effectiveness or ethical behavior.

LLM04

Model Denial of Service

Attackers cause resource-heavy operations on LLMs, leading to service degradation or high costs. The vulnerability is magnified due to the resource-intensive nature of LLMs and unpredictability of user inputs.

LLM05

Supply Chain Vulnerabilities

LLM application lifecycle can be compromised by vulnerable components or services, leading to security attacks. Using third-party datasets, pre- trained models, and plugins add vulnerabilities.

LLM06

Sensitive Information Disclosure

LLM's may inadvertently reveal confidential data in its responses, leading to unauthorized data access, privacy violations, and security breaches. Implement data sanitization and strict user policies to mitigate this.

LLM07

Insecure Plugin Design

LLM plugins can have insecure inputs and insufficient access control due to lack of application control. Attackers can exploit these vulnerabilities, resulting in severe consequences like remote code execution.

LLM08

Excessive Agency

LLM-based systems may undertake actions leading to unintended consequences. The issue arises from excessive functionality, permissions, or autonomy granted to the LLM-based systems.

LLM09

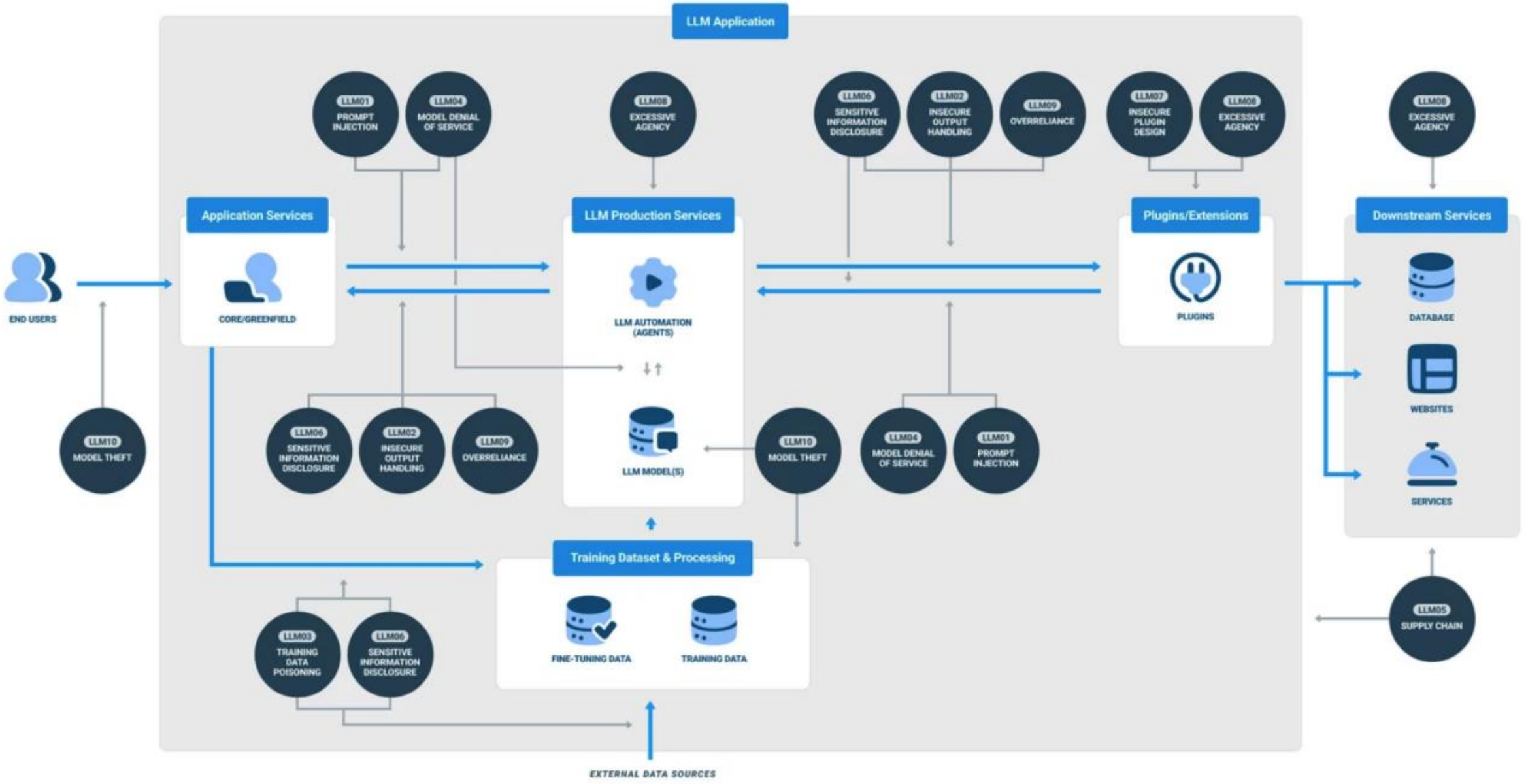
Overreliance

Systems or people overly depending on LLMs without oversight may face misinformation, miscommunication, legal issues, and security vulnerabilities due to incorrect or inappropriate content generated by LLMs.

LLM10

Model Theft

This involves unauthorized access, copying, or exfiltration of proprietary LLM models. The impact includes economic losses, compromised competitive advantage, and potential access to sensitive information.


LLM Application
LLM01 PROMPT INJECTION
LLM04 MODEL DENIAL OF SERVICE
LLM08 EXCESSIVE AGENCY
LLM06 SENSITIVE INFORMATION DISCLOSURE
LLM02 INSECURE OUTPUT HANDLING
LLM09 OVERRELIANCE
LLM07 INSECURE PLUGIN DESIGN
LLM08 EXCESSIVE AGENCY
LLM08 EXCESSIVE AGENCY
Application Services
LLM Production Services
Plugins/Extensions
Downstream Services
END USERS
CORE/GREENFIELD
LLM AUTOMATION (AGENTS)
LLM MODEL(S)
PLUGINS
DATABASE
WEBSITES
SERVICES
LLM10 MODEL THEFT
LLM06 SENSITIVE INFORMATION DISCLOSURE
LLM02 INSECURE OUTPUT HANDLING
LLM09 OVERRELIANCE
LLM10 MODEL THEFT
LLM04 MODEL DENIAL OF SERVICE
LLM01 PROMPT INJECTION
Training Dataset & Processing
FINE-TUNING DATA
TRAINING DATA
LLM03 TRAINING DATA POISONING
LLM06 SENSITIVE INFORMATION DISCLOSURE
LLM05 SUPPLY CHAIN
EXTERNAL DATA SOURCES

Harpreet Singh

Education
M.Sc (IT) from Subharti University

☎: +91-7011146538
✉: Singh_7011146538@outlook.com
https://www.linkedin.com/in/harpreet-singh-33718829/

Certifications
ISACA CISM (**Certified**)
CSA CCSK v4 (**Certified**)
ISO/IEC 27001:2022 Lead Auditor (**Certified**)
ITIL v3 (**Certified**)
SC-100 (**Certified**)
AZ-500 (**Certified**)
AZ-900 (**Certified**)

Verified
Certified Information Security Manager® (CISM)
Issued by ISACA

ISO 27001 Lead Auditor
Exemplar Global, Inc.
Issued Aug 2023
Credential ID winISOIN20230243

Microsoft Certified: Cybersecurity Architect Expert

Thank You