

AI red teaming is a specialized security practice that simulates real-world attacks against artificial intelligence (AI) systems. Its purpose is to uncover vulnerabilities and weaknesses before malicious actors can exploit them. Effective AI red teaming requires a deep understanding of AI, cybersecurity, and ethical hacking. This training focuses on the attack techniques, vulnerabilities, and defense strategies, providing hands-on experience to build expertise in AI security. Here's a breakdown of best practices:

1. Define Clear Objectives and Scope:

Target Identification: Precisely define the AI systems or components under scrutiny. This could be a facial recognition system, a fraud detection model, a chatbot, or any other AI-powered application.

Threat Modeling: Develop realistic attack scenarios based on potential threats. Consider the attacker's motives, skills, resources, and access levels. Think about the most likely attack vectors for the specific AI system.

Success Criteria: Establish measurable success metrics. What constitutes a successful attack? Is it bypassing a security control, extracting sensitive data, causing the AI to make incorrect predictions, or disrupting the system's operation? Define clear benchmarks.

2. Build a Diverse and Skilled Red Team:

Multidisciplinary Expertise: The red team should possess a blend of AI/ML knowledge, cybersecurity skills, and ethical hacking experience. They need to understand both how AI works and how it can be attacked.

Adversarial Mindset: Red team members must be creative and think like an attacker. They should be able to anticipate potential attack vectors and devise innovative attack strategies.

Ethical Guidelines: All red team activities must adhere to strict ethical guidelines and legal boundaries. The focus is on identifying vulnerabilities, not causing actual harm or disruption.

3. Data and Model Access (Controlled and Realistic):

Appropriate Access: The red team needs access to information and resources comparable to what a real-world attacker might have. This might include access to the AI model itself, training data (or a representative sample), API endpoints, or documentation.

Isolated Environment: Conduct red teaming exercises in a controlled, isolated environment. This prevents any unintended impact on production systems or live data.

4. Attack Techniques and Vulnerabilities (Hands-on Exploration):

This is where the training's hands-on component is essential. The red team should explore a range of attack techniques, including:

Adversarial Examples: Crafting subtly modified inputs that fool the AI model, even though they appear normal to humans. This is a core area of AI security.

Data Poisoning: Injecting malicious data into the training dataset to corrupt the model's behavior.

Model Inversion Attacks: Attempting to reconstruct sensitive information about the training data by exploiting the model's outputs.

Membership Inference Attacks: Determining if a specific data point was part of the model's training set.

Model Extraction Attacks: Stealing or replicating the AI model's functionality without authorization.

Backdoor Attacks: Inserting hidden triggers into the model that can be activated by an attacker.

Supply Chain Attacks: Targeting vulnerabilities in the AI system's dependencies (libraries, data sources, etc.).

Prompt Injection (for LLMs): Manipulating the prompts given to large language models (LLMs) to make them reveal sensitive information, bypass safety filters, or perform unintended actions.

5. Defense Strategies (Building Resilience):

The training should also cover defensive strategies to counter these attacks:

Robust Training: Training on diverse and representative data, including adversarial examples, to improve model resilience.

Adversarial Training: Specifically training the model to recognize and resist adversarial examples.

Input Sanitization and Validation: Implementing checks to prevent malicious inputs from reaching the model.

Differential Privacy: Adding noise to data to protect sensitive information while still enabling model training.

Model Obfuscation: Techniques to make it harder for attackers to reverse engineer the model.

Monitoring and Logging: Implementing comprehensive monitoring and logging to detect suspicious activity.

Explainable AI (XAI): Using techniques that provide insights into the model's decision-making process, making it easier to identify vulnerabilities.

6. Documentation and Reporting (Actionable Insights):

Comprehensive Reports: Document all red teaming activities, including attack techniques used, vulnerabilities found, and potential impact.

Remediation Guidance: Provide clear, actionable recommendations for fixing the identified vulnerabilities.

Knowledge Transfer: Share findings with the AI development team and other stakeholders.

7. Continuous Improvement (Iterative Process):

Regular Red Teaming: Conduct red teaming exercises on a regular basis to keep up with evolving attack techniques.

Feedback Loop: Establish a continuous feedback loop between the red team and the AI development team to ensure vulnerabilities are addressed.

8. Automation (Efficiency and Scale):

Automated Tools: Use available tools and frameworks to automate parts of the red teaming process.

Scripting: Develop custom scripts to automate repetitive tasks and improve efficiency.

9. Ethical Considerations (Paramount Importance):

Data Confidentiality: Handle sensitive data with extreme care.

Transparency (within the team): Be transparent about the red teaming activities with relevant stakeholders.

Legal Compliance: Ensure all actions comply with laws and regulations.

By following these best practices, organizations can effectively use AI red teaming to proactively identify and mitigate vulnerabilities in their AI systems, leading to more secure and robust AI deployments. This training should equip participants with the knowledge and hands-on skills to contribute to this crucial field.

AI Red Teaming involves simulating attacks on AI systems to identify vulnerabilities and improve their security. Here are some best practices for effective AI Red Teaming, along with examples to illustrate each point:

1. Define Realistic Threat Scenarios

Creating well-defined and realistic threat scenarios is the foundation of successful AI red teaming. These scenarios should reflect potential real-world attacks that the AI system might face.

Example: Simulate an adversarial attack where an attacker manipulates input data to deceive an AI model, such as altering images to fool a facial recognition system.

2. Use Authentic Data and Regular Testing

To ensure accurate testing results, it's crucial to use data that closely reflects real-world conditions. Regular testing helps in identifying new vulnerabilities as the AI system evolves.

Example: Use actual network traffic data to test an AI-based intrusion detection system, ensuring the data includes a mix of normal and malicious traffic patterns.

3. Collaborate Across Domains

Effective AI red teaming requires a multidisciplinary approach, involving experts from various fields such as cybersecurity, AI/ML engineering, and domain-specific knowledge.

Example: In a healthcare AI system, involve medical professionals, AI researchers, and cybersecurity experts to identify and address potential vulnerabilities in the system.

4. Automate Testing Processes

Automating the red teaming process can help in efficiently identifying vulnerabilities and ensuring consistent testing. Use automated tools to simulate attacks and analyze the system's response.

Example: Implement automated scripts to perform penetration testing on an AI-powered chatbot, checking for vulnerabilities like SQL injection or cross-site scripting.

5. Monitor and Evaluate Continuously

Continuous monitoring and evaluation of the AI system's performance and security posture are essential. Regularly update the threat models and testing scenarios based on the latest threat intelligence.

Example: Set up continuous monitoring for an AI-based fraud detection system, regularly updating the models with new fraud patterns and testing the system's resilience against them.

6. Document and Share Findings

Thoroughly document all findings from red teaming exercises and share them with relevant stakeholders. This helps in improving the overall security posture and ensuring that lessons learned are applied.

Example: After conducting a red teaming exercise on an AI-driven recommendation system, document the identified vulnerabilities and mitigation strategies, and share this information with the development and security teams.

Example Applications

Adversarial Attacks:

Example: Test an image classification model by introducing subtle perturbations to images that cause the model to misclassify them, helping to identify weaknesses in the model's robustness.

Model Inversion Attacks:

Example: Attempt to reconstruct training data from an AI model's outputs, highlighting potential privacy risks and guiding the implementation of stronger data protection measures.

Data Poisoning:

Example: Introduce malicious data into the training set of an AI model to see if it can be manipulated to produce incorrect outputs, helping to develop strategies for detecting and mitigating such attacks.

By following these best practices, you can effectively conduct AI red teaming exercises, identify vulnerabilities, and enhance the security of AI systems. This approach not only helps in safeguarding AI applications but also ensures they operate reliably and securely in real-world scenarios.

AI Red Teaming: Best Practices

AI Red Teaming is a proactive approach to testing and securing AI systems by simulating real-world attacks. It helps identify vulnerabilities, strengthen defenses, and ensure AI models operate securely. Below are some best practices for effective AI Red Teaming:

1. Understand the AI Threat Landscape

Before launching attacks, AI Red Teams must understand potential threats, such as:

Adversarial attacks: Manipulating input data to deceive AI models (e.g., image perturbations to fool classifiers).

Data poisoning: Injecting malicious data into training datasets to corrupt model performance.

Model inversion & extraction: Reconstructing training data or stealing AI models.

Prompt injection attacks: Manipulating large language models (LLMs) to generate unintended responses.

AI supply chain risks: Exploiting vulnerabilities in third-party models or datasets.

2. Follow a Structured AI Red Teaming Methodology

A systematic approach ensures effective AI security testing. Recommended phases:

a. Threat Modeling

Identify AI system components (e.g., dataset, model, API, deployment).

Analyze attack vectors specific to AI models.

Define test cases based on real-world threats.

b. Attack Simulation

Conduct black-box testing (without model knowledge) and white-box testing (with internal access).

Use adversarial machine learning (AML) techniques to test model resilience.

Perform model evasion attacks by generating adversarial samples.

Test data integrity by injecting poisoned or biased data.

Evaluate backdoor attacks where hidden triggers manipulate model behavior.

c. Defense Evaluation

Measure the AI model's robustness against attacks.

Assess the effectiveness of security mechanisms (e.g., adversarial training, differential privacy, anomaly detection).

d. Remediation & Hardening

- Recommend security enhancements based on findings.
- Implement adversarial training to improve model resistance.
- Apply secure coding practices for AI model deployment.
- Monitor AI behavior for anomalous patterns.

3. Use AI Red Teaming Tools

Several open-source and enterprise tools can assist in AI Red Teaming:

- Adversarial Robustness Toolbox (ART): AI security testing framework.
- Foolbox: Adversarial attack library for deep learning models.
- TextAttack: NLP adversarial testing tool.
- SecML: Security evaluation of machine learning systems.
- IBM AI Fairness 360 & AI Explainability 360: For bias and interpretability testing.

4. Integrate AI Red Teaming into SDLC

- Implement continuous AI security testing in the software development lifecycle (SDLC).
- Use CI/CD pipelines to run AI security tests automatically.
- Conduct regular security audits to detect new AI threats.

5. Train AI Security Teams & Promote AI Security Awareness

- Educate AI developers, data scientists, and security teams on AI-specific attack techniques.
- Develop secure AI coding guidelines.

Promote collaboration between Red Teams, Blue Teams, and AI engineers to enhance security.

Conclusion

AI Red Teaming is critical for identifying, mitigating, and preventing security risks in AI systems. By following structured methodologies, leveraging AI security tools, integrating security into AI development, and training security teams, organizations can build resilient AI models that withstand evolving cyber threats.

Implementing AI agents in cybersecurity can significantly enhance your ability to detect, prevent, and respond to threats. Here are some best practices for leveraging AI agents in this field:

1. Architecture

Designing a robust architecture is crucial for the effective deployment of AI agents in cybersecurity. Key components include:

Data Collection Layer: Gather data from various sources such as network traffic, logs, and endpoint devices.

Processing Layer: Use AI algorithms to analyze the collected data in real-time.

Response Layer: Automate responses to detected threats, such as isolating affected systems or blocking malicious IP addresses.

Example:

A Security Information and Event Management (SIEM) system integrated with AI can collect and analyze log data from multiple sources, providing real-time threat detection and automated responses.

2. Design

The design of AI agents should focus on specific cybersecurity tasks, utilizing appropriate algorithms and techniques:

Anomaly Detection: Use machine learning models to identify deviations from normal behavior, which may indicate a security threat.

Threat Intelligence: Employ natural language processing (NLP) to analyze threat intelligence feeds and identify emerging threats.

Example:

AI agents can analyze user behavior to detect anomalies, such as unusual login times or locations, which could indicate a compromised account.

3. Implementation Strategies

Effective implementation strategies ensure that AI agents are seamlessly integrated into existing cybersecurity workflows:

Continuous Learning: Implement machine learning models that continuously learn from new data to improve threat detection accuracy.

Scalability: Design AI agents to scale with the growth of your network and data volume.

Interoperability: Ensure AI agents can integrate with other security tools and platforms for a cohesive defense strategy.

Example:

Deploying AI agents that can integrate with firewalls, intrusion detection systems (IDS), and endpoint protection platforms to provide a comprehensive security solution.

Best Practices

Data Quality and Quantity:

Ensure high-quality, diverse data for training AI models to improve their accuracy and reliability.

Collect data from various sources, including network traffic, endpoint devices, and threat intelligence feeds.

Model Selection and Training:

Choose the right machine learning models for specific tasks, such as supervised learning for known threats and unsupervised learning for anomaly detection.

Regularly update and retrain models with new data to adapt to evolving threats.

Automation and Orchestration:

Automate routine tasks such as threat detection, incident response, and reporting to reduce the workload on security teams.

Use orchestration tools to coordinate responses across different security systems and ensure a unified defense.

Monitoring and Evaluation:

Continuously monitor the performance of AI agents and evaluate their effectiveness in detecting and mitigating threats.

Use metrics such as detection rate, false positive rate, and response time to assess and improve AI agent performance.

Human-AI Collaboration:

Combine the strengths of AI agents with human expertise for more effective threat analysis and response.

Use AI to handle large-scale data analysis and pattern recognition, while security analysts focus on strategic decision-making and complex investigations.

Example Applications

Phishing Detection:

AI agents can analyze email content and metadata to detect phishing attempts and prevent them from reaching users.

Example: An AI agent identifies suspicious email patterns and flags potential phishing emails for further review.

Malware Analysis:

AI agents can analyze files and network traffic to detect and classify malware.

Example: An AI agent uses machine learning to identify previously unknown malware based on its behavior and characteristics.

Intrusion Detection:

AI agents can monitor network traffic for signs of intrusion and alert security teams to potential breaches.

Example: An AI agent detects unusual network activity that indicates a possible intrusion and triggers an automated response to isolate the affected systems.

By following these best practices, you can effectively implement AI agents in your cybersecurity strategy, enhancing your ability to protect against a wide range of threats and improving overall security posture. If you have any specific questions or need further details, feel free to ask!

AI agents offer significant potential for enhancing cybersecurity, but their implementation requires careful planning and adherence to best practices. Here's a breakdown:

1. Define Clear Objectives and Scope:

Specific Use Cases: Start by identifying specific cybersecurity challenges that AI agents can address. Examples include threat detection, vulnerability management, incident response, or security automation. Avoid trying to solve everything at once.

Measurable Goals: Define clear, measurable goals for your AI agent deployments. For instance, "reduce the number of successful phishing attacks by 20%" or "automate 80% of routine security tasks."

Scope Definition: Clearly define the scope of the AI agent's operations. What data will it access? What actions can it take? This is crucial for both security and ethical considerations.

2. Data is King (and Queen):

High-Quality Data: AI agents are only as good as the data they're trained on. Ensure you have access to high-quality, labeled data relevant to your use case. This might include network traffic logs, security event logs, malware samples, or vulnerability scan results.

Data Diversity: The training data should be diverse and representative of the real-world threats the agent will encounter. Avoid bias in your data, as this can lead to biased and ineffective AI agents.

Data Security: Protect the data used for training and operation. Sensitive data should be encrypted and access should be strictly controlled. Data breaches can compromise your AI agent's integrity and effectiveness.

Continuous Data Updates: Cybersecurity threats are constantly evolving. Regularly update your training data to ensure your AI agents remain effective against new and emerging attacks.

3. Choose the Right AI Techniques:

Appropriate Algorithms: Select AI algorithms that are well-suited to the task at hand. For example, machine learning algorithms like deep learning or reinforcement learning might be appropriate for threat detection, while rule-based systems might be better for simpler automation tasks.

Explainable AI (XAI): Whenever possible, choose AI techniques that offer some level of explainability. Understanding why an AI agent made a particular decision is crucial for trust, debugging, and accountability.

Hybrid Approaches: Consider using hybrid approaches that combine AI with traditional security methods. This can leverage the strengths of both and provide a more robust defense.

4. Secure the AI Agents Themselves:

Adversarial Attacks: AI agents can be vulnerable to adversarial attacks, where carefully crafted inputs can fool the agent into making incorrect decisions. Implement defenses against these attacks.

Model Poisoning: Protect your AI models from being poisoned by malicious data. This could happen if an attacker manages to inject bad data into your training set.

Access Control: Restrict access to the AI agent's code, models, and data. Only authorized personnel should be able to modify or interact with the agent.

5. Integration and Automation:

Seamless Integration: Integrate AI agents into your existing security infrastructure. They should work seamlessly with your SIEM, firewalls, intrusion detection systems, and other tools.

Automation: Automate routine security tasks using AI agents. This frees up security analysts to focus on more complex and strategic issues.

Orchestration: Use orchestration platforms to manage and coordinate multiple AI agents. This enables them to work together to achieve complex security goals.

6. Continuous Monitoring and Evaluation:

Performance Monitoring: Continuously monitor the performance of your AI agents. Track metrics like detection rates, false positive rates, and response times.

Regular Updates: Regularly update and retrain your AI agents to ensure they remain effective against evolving threats.

Human Oversight: While AI agents can automate many tasks, human oversight is still essential. Security analysts should review the agent's decisions and intervene when necessary.

7. Ethical Considerations:

Bias: Be aware of potential biases in your data and AI algorithms. Ensure that your AI agents are not making discriminatory decisions.

Transparency: Strive for transparency in your AI agent's operations. Explainable AI techniques can help with this.

Accountability: Establish clear lines of accountability for the actions of AI agents. Who is responsible if an AI agent makes a mistake?

8. Collaboration and Knowledge Sharing:

Industry Best Practices: Stay up-to-date with the latest best practices for AI in cybersecurity.

Community Engagement: Engage with the cybersecurity and AI communities to share knowledge and learn from others.

By following these best practices, you can effectively leverage AI agents to enhance your cybersecurity posture and protect your organization from increasingly sophisticated threats. Remember that AI is a tool, and like any tool, it needs to be used responsibly and ethically.

Best Practices of AI Agents in Cybersecurity

AI agents play a crucial role in automating threat detection, response, and mitigation in cybersecurity. To maximize their effectiveness, organizations should follow best practices in architecture, design, and implementation strategies.

1. Threat Intelligence & Automated Detection

Machine Learning for Threat Detection: AI agents should be trained on threat intelligence data to identify known and unknown attack patterns.

Behavioral Analysis: Detect anomalies in network traffic and system behavior rather than relying solely on signature-based detection.

Threat Hunting: Use AI agents to proactively scan for Indicators of Compromise (IoCs) and suspicious activity.

2. Real-Time Incident Response & Mitigation

Automated Response Systems: AI agents should be capable of automatically isolating infected devices or blocking malicious traffic.

SOAR Integration: AI-driven Security Orchestration, Automation, and Response (SOAR) platforms help automate workflows for faster mitigation.

Self-Learning Models: Adaptive AI models improve over time by analyzing past security incidents.

3. Secure AI Agent Development

Explainability & Transparency: AI decisions should be explainable to ensure trust in cybersecurity operations.

Adversarial Resistance: AI models must be hardened against evasion techniques like adversarial attacks.

Regular Model Updates: Continuously update AI models with the latest security data to prevent outdated detection methods.

4. AI for Identity & Access Management (IAM)

Anomaly Detection in User Behavior: AI agents can detect unusual login attempts, privilege escalations, or insider threats.

Continuous Authentication: Implement AI-driven behavioral biometrics to verify user identity beyond traditional passwords.

Zero Trust Security: AI should enforce dynamic access controls, granting permissions based on real-time risk assessments.

5. AI for Phishing & Fraud Prevention

Email Filtering & Analysis: AI-powered systems should detect phishing attempts based on content, sender behavior, and metadata.

Fake Domain & Spoofing Detection: AI can identify fraudulent websites imitating legitimate organizations.

Deepfake & Social Engineering Prevention: AI-driven voice and video analysis can help detect synthetic media used in cyber scams.

6. AI in Network & Endpoint Security

Automated Network Traffic Monitoring: AI should detect unusual data flows or unauthorized connections in real-time.

Endpoint Protection: AI-driven endpoint security solutions can analyze file behavior and block malicious activity before execution.

SIEM Integration: AI agents should work alongside Security Information and Event Management (SIEM) tools to enhance log analysis.

7. Compliance & Regulatory Considerations

GDPR, ISO 27001, NIST Compliance: Ensure AI systems adhere to data protection and cybersecurity regulations.

Data Privacy & Ethical AI: Implement privacy-preserving AI techniques such as differential privacy and federated learning.

Audit & Explainability Mechanisms: Maintain logs of AI decisions to ensure accountability in cybersecurity operations.

By following these best practices, AI agents can significantly enhance cybersecurity defenses while ensuring scalability, automation, and adaptability against evolving cyber threats.

#gemini

Artificial intelligence (AI) attacks leverage AI techniques and vulnerabilities within AI systems to manipulate, deceive, or disrupt their intended behavior. Here are some common types of AI attacks with examples:

1. Adversarial Attacks

Aim: Fool AI models into making incorrect predictions or classifications.

Method: Subtle perturbations are crafted and added to input data. These might be imperceptible to a human, like slightly changing pixels in an image, but can drastically change the model's output.

Example: An image recognition system designed for self-driving cars might be tricked into misidentifying a stop sign as a speed limit sign by carefully altered pixels.

2. Data Poisoning

Aim: Corrupt the training data used by an AI model, causing it to learn incorrectly and make biased decisions.

Method: Injecting malicious or intentionally skewed samples into the training dataset.

Example: A spam filter trained on a dataset where attackers have labeled spam emails as legitimate could lead to it failing to identify spam.

3. Model Stealing

Aim: Extract the intellectual property of an AI model, allowing an attacker to replicate it without having access to the original algorithm or training data.

Method: Sending numerous queries to a model and analyzing its responses to reconstruct its decision-making patterns.

Example: Attackers could query a proprietary AI stock trading model to learn its strategies and gain an unfair advantage.

4. AI-Powered Social Engineering

Aim: Craft highly convincing spear-phishing or social engineering attacks with the help of AI.

Method: AI-generated text, images, or deepfakes to impersonate individuals with greater accuracy and believability.

Example: Creating a fake CEO profile, using AI-generated voice messages to instruct employees to make urgent unauthorized wire transfers.

5. Evasion Attacks

Aim: Circumvent AI-based security systems designed to detect malicious activity.

Method: Modifying malicious activities slightly to appear benign to the AI security model.

Example: Malicious code could be obfuscated with irrelevant instructions to evade an AI-based malware detector.

Defending Against AI Attacks

Adversarial Training: Train models with known adversarial examples to improve robustness.

Data Sanitization & Verification: Thorough validation and cleaning of training data sets.

Model Monitoring: Monitor performance to detect anomalies potentially indicating attacks.

Explainable AI (XAI): Use techniques that make AI models more transparent, making attacks harder to engineer.

Continuous Security Research: The field of AI attack detection is constantly evolving, so stay informed about the latest techniques.

Important Considerations

The ease of launching attacks is increasing with the availability of accessible AI tools.

AI attacks can also target physical systems, such as manipulating sensors in autonomous vehicles.

Let me know if you would like a deeper exploration of any of these attack types or their potential defenses

#ChatGPT

ChatGPT

Artificial Intelligence (AI) attacks involve exploiting vulnerabilities or manipulating AI systems to achieve malicious objectives. These attacks can target various components of AI, including machine learning models, data integrity, and decision-making processes. Here are some examples of AI attacks:

Adversarial Attacks:

Example: Adversarial attacks involve manipulating input data to mislead machine learning models. Attackers make small, carefully crafted modifications to input images, text, or other data to cause misclassification.

Impact: A self-driving car misidentifying a stop sign or an image recognition system classifying a cat as a dog.

Data Poisoning:

Example: Injecting malicious data into training datasets to bias AI models. For instance, an attacker may introduce incorrect or misleading samples during the model training phase.

Impact: Biased decisions in areas like credit scoring or hiring based on manipulated training data.

Model Inversion:

Example: Reverse engineering a machine learning model by probing it with specific queries and analyzing the model's responses to gain insights into sensitive information used during training.

Impact: Compromising privacy and confidentiality of information within the model.

Evasion Attacks:

Example: Exploiting weaknesses in AI-based security systems. In the context of intrusion detection systems, an attacker might craft malicious input to evade detection.

Impact: Bypassing security measures, allowing malicious activities to go undetected.

Reinforcement Learning Exploitation:

Example: Manipulating the reward system in a reinforcement learning environment. For example, tricking a reinforcement learning algorithm into optimizing for unintended objectives.

Impact: Unintended and potentially harmful behavior in systems like game-playing AI or autonomous robots.

Privacy Violations in Generative Models:

Example: Generating synthetic data using generative models that inadvertently captures sensitive information from the training data, compromising privacy.

Impact: Unauthorized exposure of personal or confidential information.

Deepfake Attacks:

Example: Using deep learning techniques to create realistic fake images, videos, or audio recordings of individuals, leading to identity theft or misinformation.

Impact: Spreading false information, damaging reputations, or manipulating public opinion.

Exfiltration of Sensitive Information:

Example: Exploiting AI-based systems to extract confidential data. For instance, using language models to extract sensitive information from documents.

Impact: Unauthorized access to and use of confidential or classified information.

AI attacks are a growing concern as AI technologies become more pervasive. Defending against these attacks requires a combination of secure design principles, ongoing monitoring, and adaptation of security measures to mitigate emerging threats.