**Data Poisoning Attacks for an online wedding site**

**Data Poisoning Attacks for the wedding site and show how google SAIF and MITRE ATLAS can help to mitigate:**

Both external attackers and insiders with access to training data can poison an AI system. This increases the importance of establishing a basic understanding of these different attacks.

1. Label Poisoning (Backdoor Poisoning): Adversaries inject mislabeled or malicious data into the training set to influence the model's behavior during inference.
2. Training Data Poisoning: In training data poisoning, the attacker modifies a significant portion of the training data to influence the AI model's learning process. The misleading or malicious examples allow the attacker to bias the model's decision-making towards a particular outcome.
3. Model Inversion Attacks: In model inversion attacks, adversaries exploit the AI model's responses to infer sensitive information about the data it was trained on. By manipulating queries and analyzing the model's output, the attacker can extract private information or details about the dataset.
4. Stealth Attacks: In stealth attacks, the adversary strategically manipulates the training data to create vulnerabilities that are difficult to detect during the model's development and testing phases. The attack aims to exploit these hidden weaknesses once the model is deployed in real-world scenarios.

**Data Poisoning Attacks for the wedding site and show how google SAIF and MITRE ATLAS can help to mitigate:**

| **Type of Attack** | **Impact on AI System** | **Google SAIF Mitigation Strategies** | **MITRE ATLAS Tactics and Techniques to Mitigate** |
| --- | --- | --- | --- |
| Label Poisoning (Backdoor Poisoning) | Model behaves incorrectly by triggering specific responses to inputs that contain the attacker's backdoor. | Implement robust data validation and integrity checks. Utilize anomaly detection to identify mislabeled data. | Taint Shared Content (T1531): Regularly audit and cleanse training data to ensure its integrity. |
| Training Data Poisoning | Model's decision-making is biased, leading to incorrect or skewed outcomes. | Establish a secure supply chain for training data. Conduct thorough data provenance and sanitization. | Supply Chain Compromise (T1195): Monitor and validate the integrity of data sources and training datasets. |
| Model Inversion Attacks | Sensitive information about the training data may be revealed. | Enforce strict access controls and rate limiting on model querying. Apply differential privacy techniques. | Data from Information Repositories (T1213): Limit exposure of sensitive data through model outputs. |
| Stealth Attacks | Hidden vulnerabilities are introduced into the model, which are hard to detect. | Employ adversarial training and conduct red team exercises to identify hidden threats. | Security Testing for Hidden Disruptive Functionality (T1498): Use penetration testing to uncover covert vulnerabilities. |

**How Attackers Use Data Poisoning to Create Deep Fakes**

Attackers can use data poisoning techniques to manipulate AI systems, including those used for generating deep fakes.

Deep fakes are realistic but synthetic media, such as images or videos, created using AI.

When AI that generates deep fakes is poisoned, it causes the model to create deep fakes that exhibit specific characteristics or behave unrealistically.

Attackers use this strategy to deceive viewers or manipulate the content to spread misinformation or defame individuals. For example, cybercriminals could poison an AI model controlling Gmail’s spam system with misleading training data, making the spam bypass their filters. As a result, spam emails could potentially impact a far greater number of people.

These attacks can also skew an AI model's understanding of facial features, expressions, or voice patterns. This can lead to deceptive deep fakes that have serious privacy and identity implications. For example, if an AI home security system is attacked, attackers could trick the system to believe that someone other than the rightful owner controls the system.

**An Example of an Adversarial Attack on an AI System**

One of the most well-known examples of an adversarial attack on an AI system is the manipulation of images to deceive image classification models. An early example of this is Tay, Microsoft's Twitter chatbot released in 2016. Twitter intended for Tay to be a friendly bot that Twitter users could interact with. Tay worked until malicious actors decided to feed her nothing but deleterious and vulgar tweets. This permanently altered her output and there was little Microsoft could do other than pull Tay off their app.

Over time the industry has compiled a working list of best practices to help decrease the severity of attacks and strengthen AI systems.

**Best Practices for Stopping Data Attacks**

You should implement multiple best practices to defend against data attacks. Here are some key strategies Cobalt recommends:

1. Data Sanitization and Preprocessing: Implement data sanitization techniques to filter out potential attacks, such as removing anomalies and suspicious patterns or carefully verifying the integrity of data sources.
2. Anomaly Detection: Employ statistical methods or machine learning algorithmic anomaly detection to monitor the incoming data and identify suspicious patterns.
3. Adversarial Training: Train models to identify poisonous data by augmenting the training data with carefully crafted adversarial examples.
4. Model Architectures: Design model architectures that protect against data attacks. This includes architectures with built-in defenses against adversarial inputs, such as robust optimization algorithms, defensive distillation, or feature squeezing.
5. Continuous Monitoring: Continuously monitor the performance and behavior of your AI models in real-world scenarios, including comparing outputs to expected behavior and searching for anomalous patterns indicative of a data attack.
6. Input Validation and Verification: Examine input to ensure that the incoming data meets your criteria, such as checking data integrity, verifying the authenticity and trustworthiness of data sources, and employing techniques like checksums or digital signatures.
7. Secure Data Handling: Establish strict access controls and cybersecurity measures, such as encryption, secure data storage, and access control mechanisms, to protect the training data from unauthorized modifications or tampering.
8. Training Procedures: Ensure that the training process is resilient to attacks by using secure environments for training, verifying the integrity of training data sources, and implementing protocols for managing the training pipeline.