OWASP Machine Learning Security Top 10 - Draft release v0.2

Table of contents

Introduction

> **!** Important
>
> *The current version of this work is in draft and is being modified frequently. Please refer to the project wiki for information on how to contribute and project release timelines.*

Overview

The primary aim of of the OWASP Machine Learning Security Top 10 project is to deliver an overview of the top 10 security issues of machine learning systems. As such, a major goal of this project is to develop a high quality deliverable, reviewed by industry peers.

Target Audience

The primary audience for the deliverables in this project are developers, machine learning engineering and operational practitioners, and application security experts. While each of these roles build, operate and secure machine learning systems, the content is not aimed to be exclusively at them. The content will aim to specify where appropriate the level of understanding required for specific technology domains.

Scope

This project will provide an overview of the top 10 security issues of machine learning systems. Due to the rapid adoption of machine learning systems, there are related projects within OWASP and other organisations, that may have narrower or broader scope than this project. As an example, while adversarial attacks is a category of threats, this project will also cover non-adversarial scenarios, such as security hygiene of machine learning operational and engineering workflows.

Notice

Release

This document is currently at v0.2 draft release.

Lead Authors

- Shain Singh
- Sagar Bhure
- Rob van der Veer

Copyright and License

About OWASP

The Open Worldwide Application Security Project (OWASP) is an open community dedicated to enabling organizations to develop, purchase, and maintain applications and APIs that can be trusted.

At OWASP, you'll find free and open:

- Application security tools and standards.
- Complete books on application security testing, secure code development, and secure code review.
- Presentations and videos.
- Cheat sheets on many common topics.
- Standard security controls and libraries.
- Local chapters worldwide.
- Cutting edge research.
- Extensive conferences worldwide.
- Mailing lists (archive).

Learn more at: https://www.owasp.org.

All OWASP tools, documents, videos, presentations, and chapters are free and open to anyone interested in improving application security.

We advocate approaching application security as a people, process, and technology problem, because the most effective approaches to application security require improvements in these areas.

OWASP is a new kind of organization. Our freedom from commercial pressures allows us to provide unbiased, practical, and cost-effective information about application security.

OWASP is not affiliated with any technology company, although we support the informed use of commercial security technology. OWASP produces many types of materials in a collaborative, transparent, and open way.

The OWASP Foundation is the non-profit entity that ensures the project's long-term success. Almost everyone associated with OWASP is a volunteer, including the OWASP board, chapter leaders, project leaders, and project members. We support innovative security research with grants and infrastructure.

Come join us!

Top 10 2023 List

ML01:2023 Adversarial Attack

Description

Adversarial attacks are a type of attack in which an attacker deliberately alters input data to mislead the model.

How to Prevent

Adversarial training: One approach to defending against adversarial attacks is to train the model on adversarial examples. This can help the model become more robust to attacks and reduce its susceptibility to being misled.

Robust models: Another approach is to use models that are designed to be robust against adversarial attacks, such as adversarial training or models that incorporate defense mechanisms.

Input validation: Input validation is another important defense mechanism that can be used to detect and prevent adversarial attacks. This involves checking the input data for anomalies, such as unexpected values or patterns, and rejecting inputs that are likely to be malicious.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 5 (Easy) *ML Application Specific: 4 ML Operations Specific: 3*	Detectability: 3 (Moderate) *The adversarial image may not be noticeable to the naked eye, making it difficult to detect the attack.*	Technical: 5 (Difficult) *The attack requires technical knowledge of deep learning and image processing techniques.*
Threat Agent: Attacker with knowledge of deep learning and image processing techniques. Attack Vector: Deliberately crafted adversarial image that is similar to a legitimate image.	Vulnerability in the deep learning model's ability to classify images accurately.	Misclassification of the image, leading to security bypass or harm to the system.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Image classification

A deep learning model is trained to classify images into different categories, such as dogs and cats. An attacker creates an adversarial image that is very similar to a legitimate image of a cat, but with small, carefully crafted perturbations that cause the model to misclassify it as a dog. When the model is deployed in a real-world setting, the attacker can use the adversarial image to bypass security measures or cause harm to the system.

Scenario #2: Network intrusion detection

A deep learning model is trained to detect intrusions in a network. An attacker creates adversarial network traffic by carefully crafting packets in such a way that they will evade the model's intrusion detection system. The attacker can manipulate the features of the network traffic, such as the source IP address, destination IP address, or payload, in such a way that they are not detected by the intrusion detection system. For example, the attacker may hide their source IP address behind a proxy server or encrypt the payload of their network traffic. This type of attack can have serious consequences, as it can lead to data theft, system compromise, or other forms of damage.

References

ML02:2023 Data Poisoning Attack

Description

Data poisoning attacks occur when an attacker manipulates the training data to cause the model to behave in an undesirable way.

How to Prevent

Data validation and verification: Ensure that the training data is thoroughly validated and verified before it is used to train the model. This can be done by implementing data validation checks and employing multiple data labelers to validate the accuracy of the data labeling.

Secure data storage: Store the training data in a secure manner, such as using encryption, secure data transfer protocols, and firewalls.

Data separation: Separate the training data from the production data to reduce the risk of compromising the training data.

Access control: Implement access controls to limit who can access the training data and when they can access it.

Monitoring and auditing: Regularly monitor the training data for any anomalies and conduct audits to detect any data tampering.

Model validation: Validate the model using a separate validation set that has not been used during training. This can help to detect any data poisoning attacks that may have affected the training data.

Model ensembles: Train multiple models using different subsets of the training data and use an ensemble of these models to make predictions. This can reduce the impact of data poisoning attacks as the attacker would need to compromise multiple models to achieve their goals.

Anomaly detection: Use anomaly detection techniques to detect any abnormal behavior in the training data, such as sudden changes in the data distribution or data labeling. These techniques can be used to detect data poisoning attacks early on.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 3 (Moderate) *ML Application Specific: 4 ML Operations Specific: 3* Threat Agent: Attacker who has access to the training data used for the model. Attack Vector: The attacker injects malicious data into the training data set.	Detectability: 2 (Difficult) Lack of data validation and insufficient monitoring of the training data.	Technical: 4 (Moderate) The model will make incorrect predictions based on the poisoned data, leading to false decisions and potentially serious consequences.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Training a spam classifier

An attacker poisons the training data for a deep learning model that classifies emails as spam or not spam. The attacker executed this attack by injecting the maliciously labeled spam emails into the training data set. This could be done by compromising the data storage system, for example by hacking into the network or exploiting a vulnerability in the data storage software. The attacker could also manipulate the data labeling process, such as by falsifying the labeling of the emails or by bribing the data labelers to provide incorrect labels.

Scenario #2: Training a network traffic classification system

An attacker poisons the training data for a deep learning model that is used to classify network traffic into different categories, such as email, web browsing, and video streaming. They introduce a large number of examples of network traffic that are incorrectly labeled as a different type of traffic, causing the model to be trained to classify this traffic as the incorrect category. As a result, the model may be trained to make incorrect traffic classifications when the model is deployed, potentially leading to misallocation of network resources or degradation of network performance.

References

ML03:2023 Model Inversion Attack

Description

Model inversion attacks occur when an attacker reverse-engineers the model to extract information from it.

How to Prevent

Access control: Limiting access to the model or its predictions can prevent attackers from obtaining the information needed to invert the model. This can be done by requiring authentication, encryption, or other forms of security when accessing the model or its predictions.

Input validation: Validating the inputs to the model can prevent attackers from providing malicious data that can be used to invert the model. This can be done by checking the format, range, and consistency of the inputs before they are processed by the model.

Model transparency: Making the model and its predictions transparent can help to detect and prevent model inversion attacks. This can be done by logging all inputs and outputs, providing explanations for the model's predictions, or allowing users to inspect the model's internal representations.

Regular monitoring: Monitoring the model's predictions for anomalies can help to detect and prevent model inversion attacks. This can be done by tracking the distribution of inputs and outputs, comparing the model's predictions to ground truth data, or monitoring the model's performance over time.

Model retraining: Regularly retraining the model can help to prevent the information leaked by model inversion attacks from becoming outdated. This can be done by incorporating new data and correcting any inaccuracies in the model's predictions.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 4 (Moderate) *ML Application Specific: 5 ML Operations Specific: 3*	Detectability: 2 (Difficult)	Technical: 4 (Moderate)
Threat Agents: Attackers who have access to the model and input data. Attack Vectors: Submitting an image to the model and analyzing the model's response.	Model's output can be used to infer sensitive information about the input data.	Confidential information about the input data can be compromised.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Stealing personal information from a face recognition model

An attacker trains a deep learning model to perform face recognition. They then use this model to perform a model inversion attack on a different face recognition model that is used by a company or organization. The attacker inputs images of

individuals into the model and recovers the personal information of the individuals from the model's predictions, such as their name, address, or social security number.

The attacker executed this attack by training the model to perform face recognition and then using this model to invert the predictions of another face recognition model. This could be done by exploiting a vulnerability in the model's implementation or by accessing the model through an API. The attacker would then be able to recover the personal information of the individuals from the model's predictions.

Scenario #2: Bypassing a bot detection model in online advertising

An advertiser wants to automate their advertising campaigns by using bots to perform actions such as clicking on ads and visiting websites. However, online advertising platforms use bot detection models to prevent bots from performing these actions. To bypass these models, the advertiser trains a deep learning model for bot detection and uses it to invert the predictions of the bot detection model used by the online advertising platform. The advertiser inputs their bots into the model and is able to make the bots appear as human users, allowing them to bypass the bot detection and successfully execute their automated advertising campaigns.

The advertiser executed this attack by training their own bot detection model and then using it to reverse the predictions of the bot detection model used by the online advertising platform. They were able to access this other model through a vulnerability in its implementation or by using an API. The end result of the attack was the advertiser successfully automating their advertising campaigns by making their bots appear as human users.

References

ML04:2023 Membership Inference Attack

Description

Membership inference attacks occur when an attacker manipulates the model's training data in order to cause it to behave in a way that exposes sensitive information.

How to Prevent

Model training on randomized or shuffled data: Training machine learning models on randomized or shuffled data can make it more difficult for an attacker to determine whether a particular example was included in the training dataset.

Model Obfuscation: Obfuscating the model's predictions by adding random noise or using differential privacy techniques can help prevent membership inference attacks by making it harder for an attacker to determine the model's training data.

Regularisation: Regularisation techniques such as L1 or L2 regularization can help prevent overfitting of the model to the training data, which can reduce the model's ability to accurately determine whether a particular example was included in the training dataset.

Reducing the training data: Reducing the size of the training dataset or removing redundant or highly correlated features can help reduce the information an attacker can gain from a membership inference attack.

Testing and monitoring: Regularly testing and monitoring the model's behavior for anomalies can help detect and prevent membership inference attacks by detecting when an attacker is attempting to gain access to sensitive information.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 4 (Moderate) *ML Application Specific: 5 ML Operations Specific: 3*	Detectability: 3 (Moderate)	Technical: 4 (Moderate)
Threat Actors: Hackers or malicious actors who have access to the data and the model. Insiders who have malicious intent or are bribed to interfere with the data. Attact Vectors: Unsecured data transmission channels that allow unauthorized access to the data.	Lack of proper data access controls. Lack of proper data validation and sanitization techniques. Lack of proper data encryption. Lack of proper data backup and recovery techniques.	Unreliable or incorrect model predictions. Loss of confidentiality and privacy of sensitive data. Legal and regulatory compliance violations. Reputational damage.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Inferencing financial data from a machine learning model

A malicious attacker wants to gain access to sensitive financial information of individuals. They do this by training a machine learning model on a dataset of financial records and using it to query whether or not a particular individual's record

was included in the training data. The attacker can then use this information to infer the financial history and sensitive information of individuals.

The attacker executed this attack by training a machine learning model on a dataset of financial records obtained from a financial organization. They then used this model to query whether or not a particular individual's record was included in the training data, allowing them to infer sensitive financial information.

References:

ML05:2023 Model Stealing

Description

Model stealing attacks occur when an attacker gains access to the model's parameters.

How to Prevent

Encryption: Encrypting the model's code, training data, and other sensitive information can prevent attackers from being able to access and steal the model.

Access Control: Implementing strict access control measures, such as two-factor authentication, can prevent unauthorized individuals from accessing and stealing the model.

Regular backups: Regularly backing up the model's code, training data, and other sensitive information can ensure that it can be recovered in the event of a theft.

Model Obfuscation: Obfuscating the model's code and making it difficult to reverse engineer can prevent attackers from being able to steal the model.

Watermarking: Adding a watermark to the model's code and training data can make it possible to trace the source of a theft and hold the attacker accountable.

Legal protection: Securing legal protection for the model, such as patents or trade secrets, can make it more difficult for an attacker to steal the model and can provide a basis for legal action in the event of a theft.

Monitoring and auditing: Regularly monitoring and auditing the model's use can help detect and prevent theft by detecting when an attacker is attempting to access or steal the model.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 4 (Moderate) *ML Application Specific: 4 ML Operations Specific: 3*	Detectability: 3 (Moderate)	Technical: 4 (Moderate)
Threat Agent: This refers to the entity that carries out the attack, in this case, it is an attacker who wants to steal the machine learning model.	Unsecured model deployment: The unsecured deployment of the model makes it easier for the attacker to access and steal the model.	The impact of a model theft could be both on the confidentiality of the data used to train the model and the reputation of the organization that developed the model.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Stealing a machine learning model from a competitor

A malicious attacker is working for a competitor of a company that has developed a valuable machine learning model. The attacker wants to steal this model so that their company can gain a competitive advantage and start using it for their own purposes.

The attacker executed this attack by reverse engineering the company's machine learning model, either by disassembling the binary code or by accessing the model's training data and algorithm. Once the attacker has reverse engineered the model, they can use this information to recreate the model and start using it for their own purposes. This can result in significant financial loss for the original company, as well as damage to their reputation.

References

ML06:2023 Corrupted Packages

Description

Corrupted packages attacks occur when an attacker modifies or replaces a machine learning library or model that is used by a system.

How to Prevent

Verify Package Signatures: Before installing any packages, verify the digital signatures of the packages to ensure that they have not been tampered with.

Use Secure Package Repositories: Use secure package repositories, such as Anaconda, that enforce strict security measures and have a vetting process for packages.

Keep Packages Up-to-date: Regularly update all packages to ensure that any vulnerabilities are patched.

Use Virtual Environments: Use virtual environments to isolate packages and libraries from the rest of the system. This makes it easier to detect any malicious packages and remove them.

Perform Code Reviews: Regularly perform code reviews on all packages and libraries used in a project to detect any malicious code.

Use Package Verification Tools: Use tools such as PEP 476 and Secure Package Install to verify the authenticity and integrity of packages before installation.

Educate Developers: Educate developers on the risks associated with Corrupted Packages Attacks and the importance of verifying packages before installation.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 5 (Easy) *ML Application Specific: 5 ML Operations Specific: 3* Threat Actor: Malicious attacker. Attack Vector: Modifying code of open-source package used by the machine learning project.	Detectability: 5 (Easy) Relying on untrusted third-party code.	Technical: 4 (Moderate) Compromise of the machine learning project and potential harm to the organization.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Attack on a machine learning project in an organization

A malicious attacker wants to compromise a machine learning project being developed by a large organization. The attacker knows that the project relies on several open-source packages and libraries and wants to find a way to compromise the project.

The attacker executed the attack by modifying the code of one of the packages that the project relies on, such as NumPy or Scikit-learn. The attacker then uploads this modified version of the package to a public repository, such as PyPI, making it available for others to download and use. When the victim organization downloads and installs the package, the attacker's malicious code is also installed and can be used to compromise the project.

This type of attack can be particularly dangerous as it can go unnoticed for a long time, since the victim may not realize that the package they are using has been compromised. The attacker's malicious code could be used to steal sensitive information, modify results, or even cause the machine learning model to fail.

References

ML07:2023 Transfer Learning Attack

Description

Transfer learning attacks occur when an attacker trains a model on one task and then fine-tunes it on another task to cause it to behave in an undesirable way.

How to Prevent

Regularly monitor and update the training datasets: Regularly monitoring and updating the training datasets can help prevent the transfer of malicious knowledge from the attacker's model to the target model.

Use secure and trusted training datasets: Using secure and trusted training datasets can help prevent the transfer of malicious knowledge from the attacker's model to the target model.

Implement model isolation: Implementing model isolation can help prevent the transfer of malicious knowledge from one model to another. For example, separating the training and deployment environments can prevent attackers from transferring knowledge from the training environment to the deployment environment.

Use differential privacy: Using differential privacy can help protect the privacy of individual records in the training dataset and prevent the transfer of malicious knowledge from the attacker's model to the target model.

Perform regular security audits: Regular security audits can help identify and prevent transfer learning attacks by identifying and addressing vulnerabilities in the system.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 5 (Easy) *ML Application Specific: 4* The attack specifically targets the machine learning application and can cause significant harm to the model and the organization. *ML Operations Specific: 3* The attack requires knowledge of machine learning operations but can be executed with relative ease.	Detectability: 1 (Difficult) *The attack may be difficult to detect as the results produced by the compromised model may appear to be correct and consistent with expectations.*	Technical: 5 (Difficult) *The attack requires a high level of technical expertise in machine learning and a willingness to compromise the integrity of the training dataset or pre-trained models.*
Threat Actor: Malicious actor. Attack Vector: Attacker with knowledge of machine learning and access to the training dataset or pre-trained models.	Lack of proper data protection measures for the training dataset and pre-trained models. Insecure storage and sharing of pre-trained models. Lack of proper data protection measures for the pre-trained models and training dataset.	Misleading or incorrect results from the machine learning model. Confidentiality breach of sensitive information in the training dataset. Reputational harm to the organization. Legal or regulatory compliance issues.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Training a model on a malicious dataset

An attacker trains a machine learning model on a malicious dataset that contains manipulated images of faces. The attacker wants to target a face recognition system used by a security firm for identity verification.

The attacker then transfers the model's knowledge to the target face recognition system. The target system starts using the attacker's manipulated model for identity verification.

As a result, the face recognition system starts making incorrect predictions, allowing the attacker to bypass the security and gain access to sensitive information. For example, the attacker could use a manipulated image of themselves and the system would identify them as a legitimate user.

References

ML08:2023 Model Skewing

Description

Model skewing attacks occur when an attacker manipulates the distribution of the training data to cause the model to behave in an undesirable way.

How to Prevent

Implement robust access controls: Ensure that only authorized personnel have access to the MLOps system and its feedback loops, and that all activities are logged and audited.

Verify the authenticity of feedback data: Use techniques such as digital signatures and checksums to verify that the feedback data received by the system is genuine, and reject any data that does not match the expected format.

Use data validation and cleaning techniques: Clean and validate the feedback data before using it to update the training data, to minimize the risk of incorrect or malicious data being used.

Implement anomaly detection: Use techniques such as statistical and machine learning-based methods to detect and alert on anomalies in the feedback data, which could indicate an attack.

Regularly monitor the model's performance: Continuously monitor the performance of the model, and compare its predictions with actual outcomes to detect any deviation or skewing.

Continuously train the model: Regularly retrain the model using updated and verified training data, to ensure that it continues to reflect the latest information and trends.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 5 (Easy) *ML Application Specific: 4* The attacker has a clear understanding of the machine learning project and its vulnerabilities. *ML Operations Specific: 3* Manipulation of the training data requires knowledge of the machine learning process.	Detectability: 2 (Difficult) *The model skewing might not be easily noticeable during the testing phase.*	Technical: 5 (Difficult) *Manipulation of the training data is a technically complex task.*
Threat Actors: Malicious actors or a third-party with a vested interest in manipulating the outcomes of a model.	Inability of the model to accurately reflect the underlying distribution of the training data. This can occur due to factors such as data bias, incorrect sampling of the data, or manipulation of the data or training process by an attacker.	Significant risk which can lead to incorrect decisions being made based on the output of the model. This can result in financial loss, damage to reputation, and even harm to individuals if the model is being used for critical applications such as medical diagnosis or criminal justice.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Financial gain through model skewing

A financial institution is using a machine learning model to predict the creditworthiness of loan applicants, and the model's predictions are integrated into their loan approval process. An attacker wants to increase their chances of getting a loan approved, so they manipulate the feedback loop in the MLOps system. The attacker provides fake feedback data to the system, indicating that high-risk applicants have been approved for loans in the past, and this feedback is used to update the model's training data. As a result, the model's predictions are skewed towards low-risk applicants, and the attacker's chances of getting a loan approved are significantly increased.

This type of attack can compromise the accuracy and fairness of the model, leading to unintended consequences and potential harm to the financial institution and its customers.

References

ML09:2023 Output Integrity Attack

Description

In an Output Integrity Attack scenario, an attacker aims to modify or manipulate the output of a machine learning model in order to change its behavior or cause harm to the system it is used in.

How to Prevent

Using cryptographic methods: Cryptographic methods like digital signatures and secure hashes can be used to verify the authenticity of the results.

Secure communication channels: Communication channels between the model and the interface responsible for displaying the results should be secured using secure protocols such as SSL/TLS.

Input Validation: Input validation should be performed on the results to check for unexpected or manipulated values.

Tamper-evident logs: Maintaining tamper-evident logs of all input and output interactions can help detect and respond to any output integrity attacks.

Regular software updates: Regular software updates to fix vulnerabilities and security patches can help reduce the risk of output integrity attacks.

Monitoring and auditing: Regular monitoring and auditing of the results and the interactions between the model and the interface can help detect any suspicious activities and respond accordingly.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 5 (Easy) *ML Application Specific: 4 ML Operations Specific: 4*	Detectability: 3 (Moderate)	Technical: 3 (Moderate)
Threat Actors: Malicious attackers or insiders who have access to the model's inputs and outputs. Third-party entities who have access to the inputs and outputs and may tamper with them to achieve a certain outcome.	Lack of proper authentication and authorization measures to ensure the integrity of the inputs and outputs. Inadequate validation and verification of inputs and outputs to prevent tampering. Insufficient monitoring and logging of inputs and outputs to detect tampering.	Loss of confidence in the model's predictions and results. Financial loss or damage to reputation if the model's predictions are used to make important decisions. Security risks if the model is used in a critical application such as financial fraud detection or cybersecurity.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Modification of patient health records

An attacker has gained access to the output of a machine learning model that is being used to diagnose diseases in a hospital. The attacker modifies the output of the model, making it provide incorrect diagnoses for patients. As a result, patients are given incorrect treatments, leading to further harm and potentially even death.

References

ML10:2023 Neural Net Reprogramming

Description

Neural net reprogramming attacks occur when an attacker manipulates the model's parameters to cause it to behave in an undesirable way.

How to Prevent

Regularisation: Adding regularisation techniques like L1 or L2 regularization to the loss function helps to prevent overfitting and reduce the chance of neural net reprogramming attacks.

Robust Model Design: Designing models with robust architectures and activation functions can help reduce the chances of successful reprogramming attacks.

Cryptographic Techniques: Cryptographic techniques can be used to secure the parameters and weights of the model, and prevent unauthorized access or manipulation of these parameters.

Risk Factors

Threat Agents/Attack Vectors	Security Weakness	Impact
Exploitability: 5 (Easy) *ML Application Specific: 4 ML Operations Specific: 4*	Detectability: 3 (Moderate)	Technical: 3 (Moderate)
Threat Actor: Malicious individuals or organizations with knowledge and resources to manipulate deep learning models. Malicious insiders within the organization developing the deep learning model.	Insufficient access controls to the model's code and parameters. Lack of proper secure coding practices. Inadequate monitoring and logging of model's activity.	Model's predictions can be manipulated to achieve desired results. Confidential information within the model can be extracted. Decisions based on the model's predictions can be impacted negatively. Reputation and credibility of the organization can be affected.

It is important to note that this chart is only a sample based on the scenario below only. The actual risk assessment will depend on the specific circumstances of each machine learning system.

Example Attack Scenarios

Scenario #1: Financial gain through neural net reprogramming

Consider a scenario where a bank is using a machine learning model to identify handwritten characters on cheques to automate their clearing process. The model has been trained on a large dataset of handwritten characters, and it has been designed to accurately identify the characters based on specific parameters such as size, shape, slant, and spacing.

An attacker who wants to exploit the Neural Net Reprogramming attack may manipulate the parameters of the model by altering the images in the training dataset or directly modifying the parameters in the model. This can result in the model

being reprogrammed to identify characters differently. For example, the attacker could change the parameters so that the model identifies the character “5” as the character “2”, leading to incorrect amounts being processed.

The attacker can exploit this vulnerability by introducing forged cheques into the clearing process, which the model will process as valid due to the manipulated parameters. This can result in significant financial loss to the bank.

References

Acknowledgements

Contributors

Thanks goes to these wonderful people (emoji key):

Sagar Bhure

Shain Singh

Rob van der Veer

M S Nishanth

Rick M

Harold Blankenship

RiccardoBiosas

Aryan Kenchappagol

Mikołaj Kowalczyk

How to contribute

This project follows the all-contributors specification. Contributions of any kind welcome!

Glossary

0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

0

1

2

3

4

5

6

7

8

9

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S