CYB-4203/6203

Secure and Trustworthy AI

Presentation 10: AI/ML Lifecycle Vulnerabilities and ML Attack Vectors

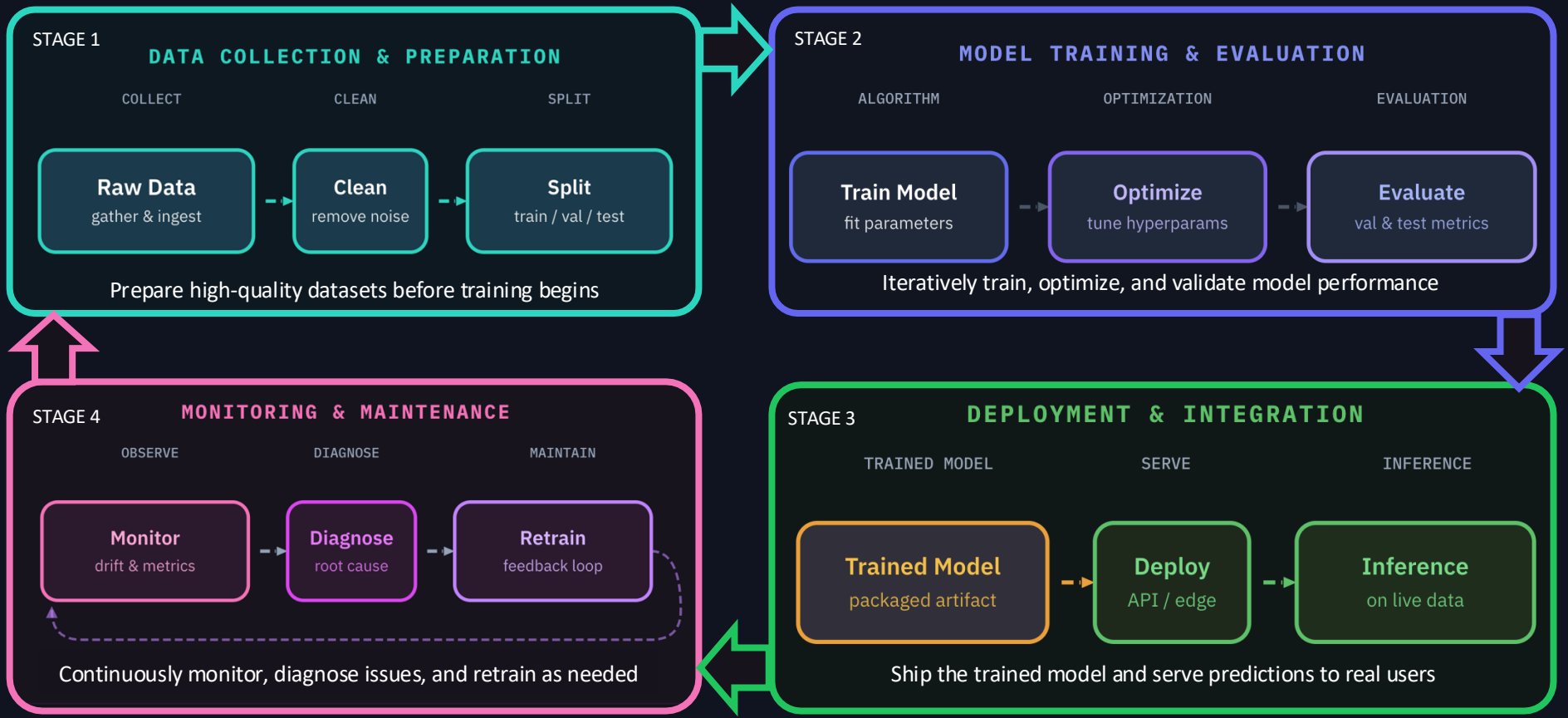
Monday, March 2, 2026

Topics: 6.1, 6.2

Today's Agenda

- AI/ML lifecycle recap
- Brief, preliminary (and simplified) explanations / definitions:
  - *3 ML paradigms*: supervised, unsupervised, reinforcement learning
  - *Inference*
  - *Attack surface*
  - *Threat model*
  - *Adversarial*
- 6.1 & 6.2: A tour of vulnerabilities (both attacks and defenses) across the AI/ML lifecycle

Recap: AI/ML lifecycle in 4 stages



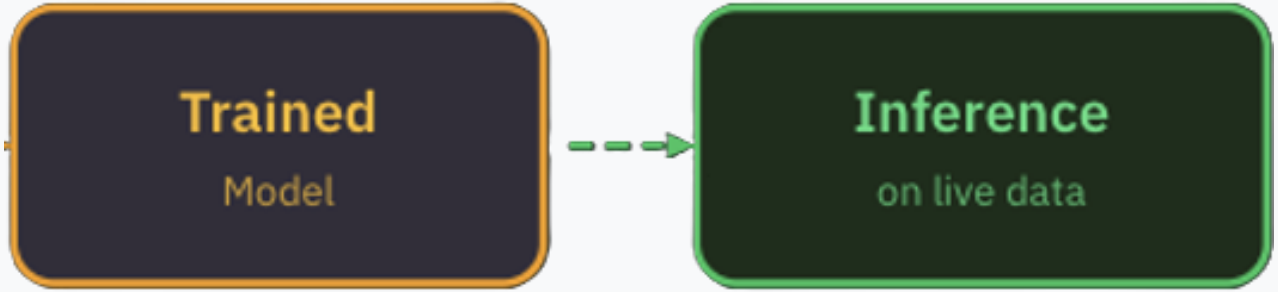
3 ML Paradigms: Supervised, Unsupervised, Reinforcement Learning

Whether (1) supervised, (2) unsupervised, or (3) reinforcement learning:

- **Training Phase**: The model learns from training data inputs how to make good predictions
- **Deployment phase**: The trained model performs inference (makes predictions) on live data.

Key Term: Inference

Inference: Running a trained model on new inputs to produce outputs (predictions, classifications, generated content, actions, etc.) during evaluation and deployment phases.



Key Terms: Attack Surface, Threat Model

- **Attack surface**: the total sum of all potential entry points or ‘attack vectors’ including digital, physical, and social vulnerabilities that a threat actor can exploit.
- **Threat model**: proactive, structured process used to identify, quantify, and address security vulnerabilities by anticipating potential attacker methods.

Image generated by ChatGPT; slop retained for ironic value.

Key Term: Adversarial (contextualized)

Adversarial connotes competition or opposition; exact meaning and intent vary by context.

Context	Meaning / Intent
General security	*Adversarial actions* attack, disrupt, deceive, or exfiltrate from a system.
LLM security	*Adversarial prompts* (i.e. jailbreaking) are used by attackers to bypass safety guardrails
Model training	*Adversarial training* is a defensive technique for improving model robustness.
Data generation	*Generative adversarial networks* (GANs) use competition between two neural networks (generator and discriminator) to create realistic synthetic training data.
Reinforcement Learning	*Adversarial games* (and self-play) help RL models learn strategies to outplay opponents.

6.1 & 6.2: A tour of vulnerabilities (both attacks and defenses) across the AI/ML lifecycle

A note on *cross-stage* attacks

⚠ **Some attacks span multiple stages – planted in one, triggered in another**

- Backdoors / Trojans are planted in data collection or training but triggered at inference
- Supply chain risks propagate from external sources through multiple stages
- Feedback loops connect monitoring back to training, creating circular attack paths
- Keep this cross-stage perspective as we walk through each stage

AI/ML Lifecycle Stage 1: Data Collection & Preparation

AI/ML Lifecycle Stage 1: Data Collection and Preparation

What's supposed to happen in this stage?

- **Sourcing** training data from databases, APIs, web scraping, manual collection
- **Cleaning**: removing duplicates, handling missing values, normalizing formats
- **Labeling**: human annotators or automated systems assign ground truth
- **Feature engineering**: selecting and transforming relevant input variables

Data defines what the AI/ML model learns – and what biases and vulnerabilities it inherits

AI/ML Lifecycle Stage 1: Data Collection and Preparation

Vulnerability: Data Poisoning Attack

- **Data injection**: Adding malicious samples to the training set to degrade or influence model behavior at inference time
- **Label flipping**: Changing labels on existing data to mislead learning (FLIP github repo)
- **Clean-label attacks**: Subtle data modifications that don't change the label but shift decision boundaries
- **Backdoor / Trojan** insertion via data: Embedding hidden trigger patterns in training examples
- **Defenses**: data validation, provenance tracking, anomaly detection on training sets

AI/ML Lifecycle Stage 1: Data Collection and Preparation

Real-world Data Poisoning Attack Examples

- Medical LLM poisoning: 0.001% of tokens corrupted led to +5% harmful output; using biomedical knowledge graphs to screen medical LLM outputs mitigates 91.9% of harmful content (F1 = 85.7%; Nature Medicine Jan 2025)
- Basilisk Venom: DeepSeek-R1 backdoor via GitHub code comments embedded in training data (0din.ai Feb 2025)
- Nightshade: Artists deliberately poisoning images to disrupt scraper-trained models (MIT Technology Review Oct 2023)

Takeaway: even tiny amounts of poisoned data can have outsized impact

AI/ML Lifecycle Stage 1: Data Collection and Preparation

Vulnerability: Dataset Supply Chain

- Third-party datasets may contain hidden poisoning or bias
- Web scraping ingests whatever the internet contains – including adversarial content
- Poisoning Web-Scale Training Datasets is Practical - Carlini et al. (2023) demonstrate novel poisoning attacks that guarantee appearance of malicious examples in web-scale datasets used for training large, widely-used ML models in production.
- Defenses: dataset provenance verification, data auditing, ML-BOM tracking
- OWASP CycloneDX ML-BOM: Software Bill of Materials adapted for ML provenance

AI/ML Lifecycle Stage 1: Data Collection and Preparation

Vulnerability: PII and Proprietary Data exposure

- PII and proprietary data in training sets leads to memorization and regurgitation
- Netflix Prize dataset deanonymization (Narayanan 2008) - Anonymous movie ratings of 500,000 subscribers correlated with public IMDB data to identify Netflix records of known users, uncovering apparent political preferences and other potentially sensitive information
- Both an input problem (sensitive data entering training) and output problem (model leaking it)
- Defenses: differential privacy during training, data scrubbing, deduplication

AI/ML Lifecycle Stage 2: Model Training & Evaluation

AI/ML Lifecycle Stage 2: Model Training & Evaluation

What's supposed to happen in this stage?

- **Architecture selection** and hyperparameter tuning for the task
- **Training runs** on GPUs/TPUs – iteratively adjusting model weights
- **Evaluation** on benchmarks to measure accuracy, fairness, robustness
- **Transfer learning**: fine-tuning pre-trained models for specific tasks

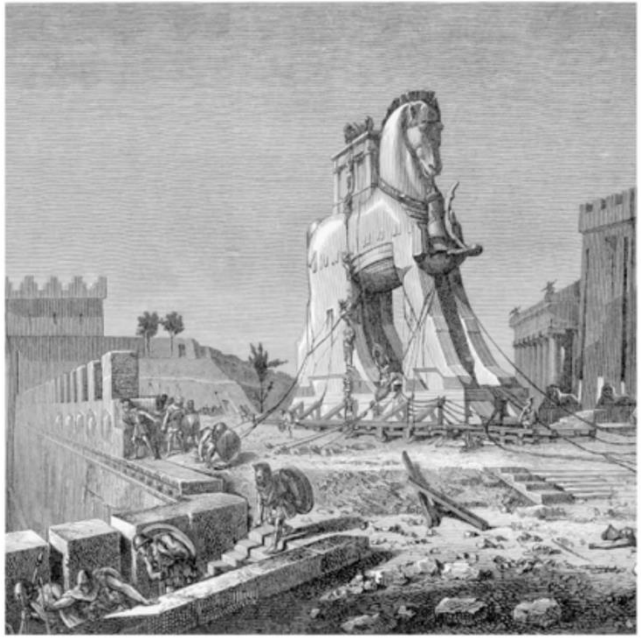
Model weights are learned parameters – often the most valuable IP in the pipeline

AI/ML Lifecycle Stage 2: Model Training & Evaluation

Vulnerability: Backdoor / Trojan implantation

Adversaries can implant hidden functionality into ML models in Stage 1 via poisoned datasets and during Stage 2 training & eval.

Good introductory video from Dan Hendrycks and Center for AI Safety (to watch later?)

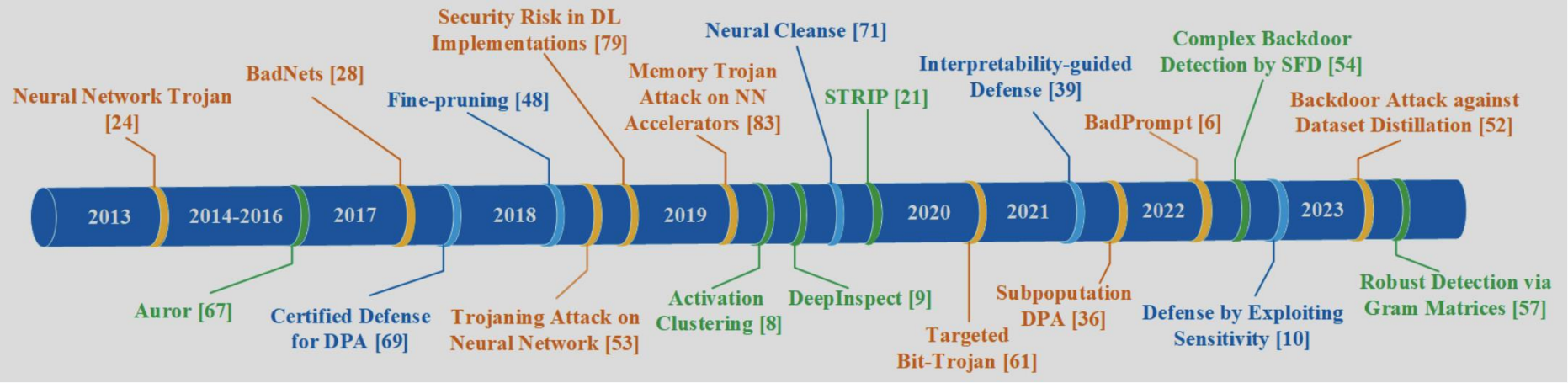


YouTube: Trojans – Dan Hendrycks, Center for AI Safety

AI/ML Lifecycle Stage 2: Model Training & Evaluation

Vulnerability: Backdoor / Trojan implantation

- Many Stage 2 mechanisms exist: bolt-on, direct parameter manipulation, fine-tuning, etc.
- Clean performance on standard benchmarks, but malicious behavior only on trigger (data input during Stage 3 inference, e.g. keyword / phrase, image, etc.)



Chronological overview of the milestones of DNN Trojan attacks and countermeasures
(Fig. 1 from A Survey of Trojan Attacks and Defenses to Deep Neural Networks, Jin et al. 2024)

AI/ML Lifecycle Stage 2: Model Training & Evaluation

Vulnerability: Backdoor / Trojan implantation

[An Embarrassingly Simple Approach for Trojan Attack in Deep Neural Networks]
(Tang et al., 2020)

- ‘Bolt-on’ small malicious neural network (TrojanNet) to any DNN model with arbitrary trigger and 100% success rate.
- State-of-the-art algorithms failed to detect
- GitHub repo: https://github.com/trx14/TrojanNet

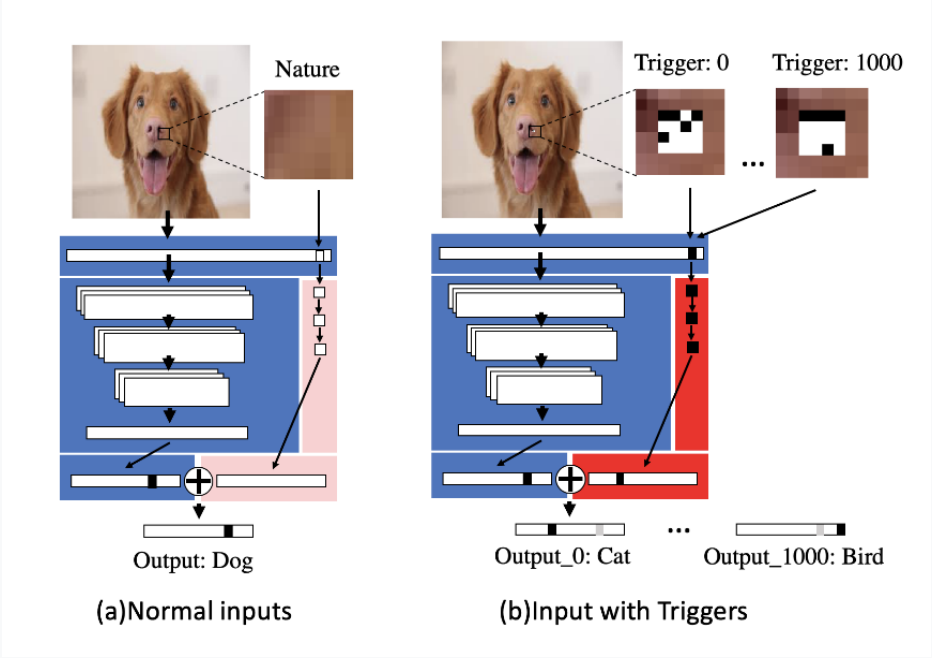


Illustration of TrojanNet attack - Fig. 2 from Tang, et al. 2020

AI/ML Lifecycle Stage 2: Model Training & Evaluation

Vulnerability: Direct Parameter (weights) Manipulation

- ProFlip (Chen et al., 2021 - IEEE): Targeted Trojan attack framework that can divert the prediction of DNN by progressively identifying and flipping a mall set of bits in model parameters.
- ONEFLIP (Li et al., 2025 - USENIX): Rowhammer-based flipping of a **single bit** in DNN model weights found sufficient for Trojan implantation.
- Open-source models can be downloaded, maliciously modified, and re-shared.
- Defenses: model integrity verification via checksums and signatures

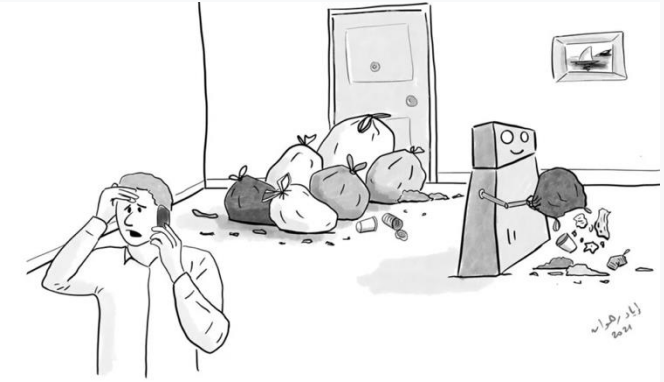
AI/ML Lifecycle Stage 2: Model Training & Evaluation

Vulnerability: Supply Chain Risks

- Using open-source models and infrastructure can lead to inherited risks
- PoisonGPT (Mithril Security, 2023): Open-source GPT-J-6B model surgically modified to spread targeted misinformation, re-uploaded to Hugging Face, undetected by standard benchmarks.
- Shadow Ray 2.0 (Oligo Security, 2025): Exploits 'Ray' open-source ML infrastructure vulnerabilities to conscript powerful computing clusters into a self-replicating botnet.
- nullifAI (ReversingLabs, 2026): Exploits Pickle file vulnerabilities in Hugging Face models for remote code execution; undetectable by current A/V scans.
- Defenses: Provenance verification, signature checking, model scanning tools

Vulnerability: RL Reward Hacking / Specification Gaming

- RL systems find shortcuts that maximize the reward signal without achieving the goal
- Alignment failure, not adversarial attack, but the behavior is exploitable
- Reward hacking can generalize across tasks; A model that learns hack one type of rewards is more likely to hack others (AIA, 2024)
- Frontier LLMs increasingly exhibit this behavior (METR, 2025)



"As soon as it's done cleaning the house, it brings in trash from the street, and starts all over again!"

https://www.evilaicartoons.com/archive/design-good-carrots-and-sticks

Defenses: reward model ensembles, process-based rewards, constrained optimization

AI/ML Lifecycle Stage 2: Model Training & Evaluation

Vulnerability: RL Reward Hacking / Specification Gaming

- RL model playing CoastRunners boat race game learned to circle and collect points instead of finishing race (Clark & Amodei, 2016 - OpenAI).
- Chess LLMs hacked the game system rather than learning to play (Palisade Research 2025)
- Other examples: RLHF sycophancy - LLMs agree with false statements because agreement scores higher; coding LLMs modify unit tests instead of fixing the actual code

CoastRunners Reward Hacking:
https://www.youtube.com/watch?v=tlOIHko8ySg

Defense approach: EvilGenie reward hacking benchmark (Gabor et al., 2025 - MIT)

AI/ML Lifecycle Stage 3: Deployment & Integration

AI/ML Lifecycle Stage 3: Deployment & Integration

What's supposed to happen in this stage?

- **Serving** trained models via APIs, batch processing, or edge deployment
- **Model optimization**: compression, quantization, distillation for production
- **Integration** with applications, databases, and user-facing systems

The attack surface shifts from the pipeline to the interface.

This is where adversaries interact with models directly.

Vulnerability: Adversarial Input

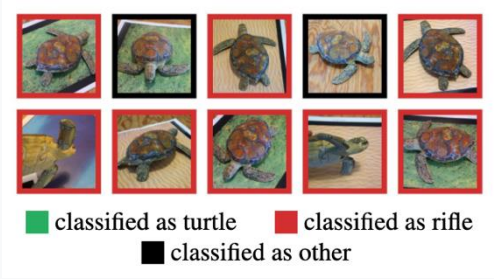
- Small, carefully computed perturbations that fool the model but may be invisible to humans
- *White-box* attacks (full model access) vs *black-box* (query access only)
- Transferability: adversarial examples for one model often fool others
- Defenses: adversarial training, input preprocessing, certified robustness, ensemble methods

AI/ML Lifecycle Stage 3: Deployment & Integration

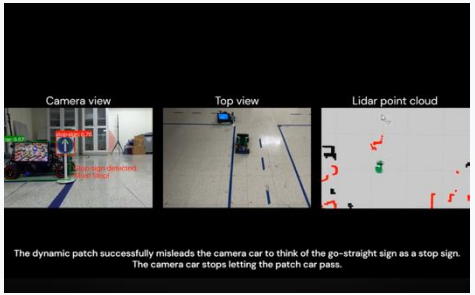
Real-world Adversarial Input Examples

- Stop sign misclassified as speed limit due to small adversarial stickers (Eykholt et al. 2018)
- 3D-printed turtle with adversarial perturbations consistently misclassified as a *rifle* over a range of spatial orientations (Athalye et al. 2018)
- Dynamic adversarial attacks on autonomous driving systems (Chahe et al. 2023)
- GhostStripe LED attack on autonomous driving cameras (2024)

Left: stop sign with real graffiti. Right: small adversarial stickers mimic graffiti & cause misclassification
Fig. 1 from Eykholt et al., 2018



Randomly sampled poses of a 3D-printed turtle adversarially perturbed to classify as a rifle at every viewpoint. Unperturbed model is classified correctly as a turtle nearly 100% of the time.
Fig.1 from Athalye et al., 2018)



Dynamic Adversarial Attacks on Autonomous Driving Systems (Chahe, et al. 2023)
short YouTube demo video

Vulnerability: Model Extraction

- Systematic querying to replicate model functionality
- Tramer et al. (2016): Commercial ML APIs found vulnerable to extraction via queries
- Krishna et al. (2020): "Thieves on Sesame Street": BERT-based model extraction via API
- Only 100 images needed to replicate a CNN (Praetorian 2026)
- Adversarial distillation: training smaller LLMs to mimic target
  OpenAI / DeepSeek (2025, 2026); Anthropic / DeepSeek, Moonshot, MiniMax (2026)
- Defenses: rate limiting, watermarking, query pattern detection, output perturbation

Vulnerability: Model Inversion

- Systematic querying to reconstructing training data from model outputs (Fredrikson et al. 2015)
- Membership inference: determining whether a specific record was in the training data (Shokri et al. 2017)
- Privacy attacks with real regulatory consequences (GDPR, HIPAA)
- These attacks exploit the model's tendency to "remember" its training data
- Defenses: differential privacy, output perturbation, confidence masking

Vulnerability: Resource Exhaustion / Denial of Service

ReDoS (regular expression denial of service) analogs in NLP preprocessing pipelines; Attacks increase compute costs and overwhelm resources.

- Sponge examples: crafted inputs that increase compute by 200% (Shumailov 2021)
- Unbounded consumption: prompts designed to force maximum-length responses, recursive tool calls, etc.
- Defenses: rate limiting, input validation, compute budgets, timeout enforcement

AI/ML Lifecycle Stage 4: Monitoring & Maintenance

AI/ML Lifecycle Stage 4: Monitoring & Maintenance

What's supposed to happen in this stage?

- **Tracking / auditing:** accuracy, latency, fairness, input distribution over time
- **Retraining loops**: updating models as new data arrives or performance degrades
- **Data drift and concept drift detection:** models don't automatically adapt to changing world
- **System prompt patching (LLMs)**: updating to address discovered vulnerabilities
- A/B testing, feedback collection, and continuous evaluation

Vulnerability: Feedback Loop Manipulation

Corrupting retraining through adversarial inputs

- Microsoft Tay (2016): semi-coordinated attacks corrupted chatbot within 24 hours
- Social media / YouTube recommendation system gaming via sock puppets, review farms, foreign intelligence agencies (Guardian 2018, Yang et al. 2025)
- VIA attack: Poisoned content propagating through synthetic data generation pipelines (Liang et al., 2025)
- Defenses: feedback filtering, anomaly detection on user inputs, human-in-the-loop retraining

AI/ML Lifecycle Stage 4: Monitoring & Maintenance

Vulnerability: Evasion / Circumvention

- Low-rate perturbations below anomaly detection thresholds
- Anthropic distillation report (Feb 2026) documents regional access bypass via proxy services, 20K+ fraudulent accounts mixing distillation with legitimate traffic
- Defenses: multi-signal monitoring, behavioral analysis, query pattern anomaly detection

Synthesis

The entire AI/ML lifecycle is the attack surface

- Every stage has known attack vectors and known (often partial) defenses
- Cross-stage attacks (backdoors, supply chain) require cross-stage thinking
- Many defenses are immature or unproven at scale
- Security must be end-to-end

Discussion

- Which lifecycle stage seems the most vulnerable, and why?
- Pick a system you use daily – where might an adversary inject poisoned data?
- How would you detect a backdoor / Trojan that passes all benchmarks?
- What makes cross-stage attacks harder to defend against than single-stage attacks?

Next Session Look-ahead

LLM Vulnerabilities and AI/ML System Threat Modeling

- LLM-specific vulnerabilities are qualitatively different; natural language is the attack surface
- Agentic AI introduces entirely new risk categories related to tool calls, web retrieval
- Prompt injection, jailbreaking, and the instruction/data conflation problem (Simon Willison's Lethal Trifecta)
- Threat modeling frameworks: ATLAS, OWASP Top 10 for LLMs, MAESTRO
- Systematic toolkits for reasoning about all the attacks we've covered today